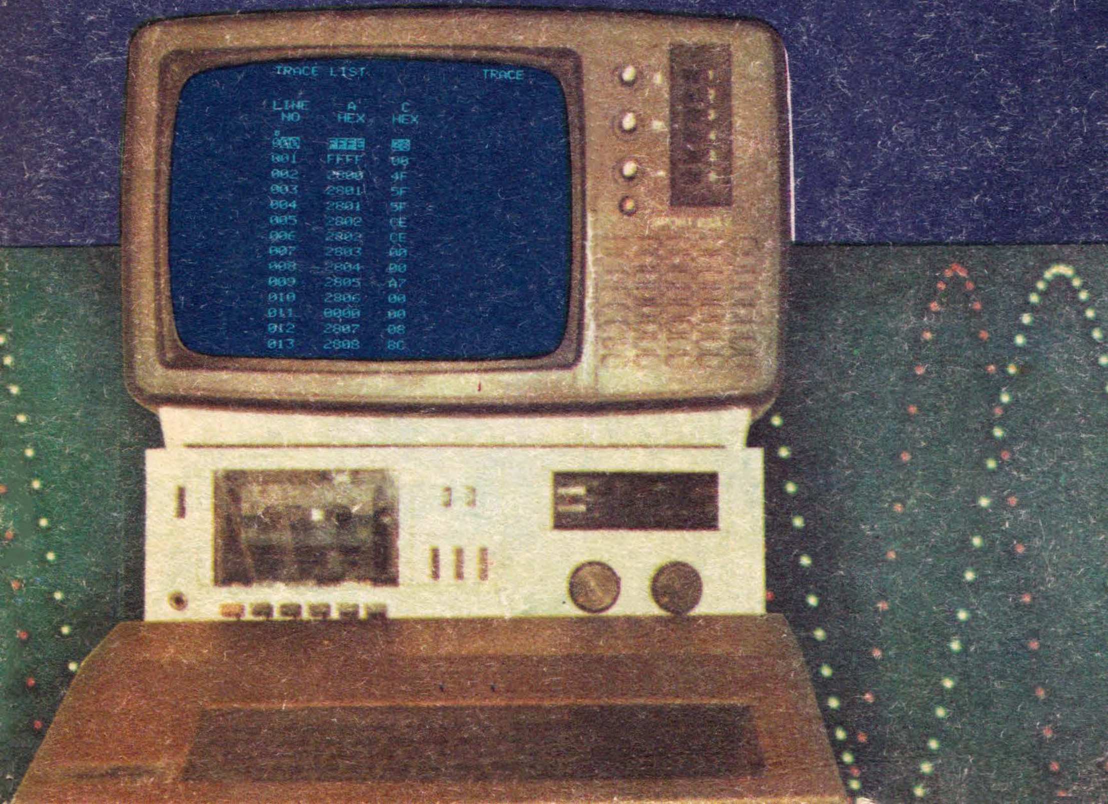
A. PETRESCU : coordonator

GH. RIZESCU
F. IACOB
C. CONSTANTINESCU
T. ILIN
E. DECSOV
C. NOVĂCESCU
A. MATEKOVITS
F. BAR
R. BERINDEANU
I. PETRESCU

Volumul 2

TOTUL DESPRE ... CALCULATORUL PERSONAL aMIC



AUTOMATICA

INFORMATICA

SERIA PRACTICA

ELECTRONICA

MANAGEMENT

BIBLIOTECA DE ● AUTOMATICĂ ● INFORMATICĂ ● ELECTRONICĂ ● MANAGEMENT

SERIA PRACTICĂ

M. K. Starr. Conducerea producţiei.
A. Vlădescu ş.a. Radioreceptoare
M. Mayer. Tiristoare în practică. Mutatoare cu comutaţie forţată
G. Möltgen. Tiristoare în practică. Mutatoare cu comutaţie de la reţea
L. Zamfirescu, I. Oprescu. Automatizarea cuptoarelor industriale
I. Papadache. Automatica aplicată, ediţia I şi a II-a
Şt. Alexandru. Automatizarea proceselor tehnologice în industria lemnului
V. H. Lisicikin. Prognoza tehnico-ştiinţifică în ramurile industriei
G. Raymond. Tehnica televiziunii în culori
T. Homoş. Capacitatea de producţie în construcţii de maşini
S. Radu, D. Filoti. Centrale telefonice automate. Sisteme de comutaţie.
R. Stere ş.a. Tranzistoare cu efect de cîmp
D. N. Sapiro. Proiectarea radioreceptoarelor
V. Antonescu, M. Popovici. Ghid pentru controlul statistic al calităţii producţiei
N. Stanciu ş.a. Tehnica imaginii în cinematografie şi televiziune
P. Vezeanu, Şt. Pătraşcu. Măsurarea temperaturii în tehnică
T. Penescu, V. Petrescu. Măsurarea presiunii în tehnică
P. Popescu, P. Mihordea. Măsurarea debitului în tehnică
P. Vezeanu. Măsurarea nivelului în tehnică
C. Hidoş, P. Isac (coordonatori) Studiul muncii, vol. I—VIII
V. Baltac ş.a. Calculatorul FELIX C-256, Structură şi programare
R. L. Morris. Proiectarea cu circuite integrate TTL
Ishikawa Kaoru. Controlul de calitate pentru maiştri şi şefi de echipe
A. M. Buhtiarov ş.a. Culegere de probleme de programare
P. Constantinescu. Sisteme informatice, modele ale conducerii şi sistemelor conduse
E. S. Buffa. Conducerea modernă a producţiei, vol. I şi II
A. Vătăşescu ş.a. Dispozitive semiconductoare. Manual de utilizare
A. Nadolo. Măsurarea volumului şi cantităţii lichidelor în industrie
Ch. Jones. Design. Metode şi aplicaţii
Gh. Pisău ş.a. Elaborarea şi introducerea sistemelor informatice
C. Hidoş. Analiza şi proiectarea circuitelor informaţionale în unităţile economice
A. Vătăşescu ş.a. Circuite integrate liniare. Manual de utilizare vol. *1, 2, 3, 4*
M. Silişteanu ş.a. Scheme de televizoare, magnetofoane, picupuri vol. *1* şi *2* ed. a II-a
D. W. Davies. Reţele de interconectarea calculatoarelor
V. Pescaru ş.a. Fişiere, baze şi bănci de date
Gh. Baştiurea ş.a. Comanda numerică a maşinilor-unelte
N. Sprînceană ş.a. Automatizări discrete în industrie. Culegere de probleme
M. Florescu. Cibernetică, automatică, informatică în industria chimică
S. Călin. Optimizări în automatizări industriale
S. Maican. Sisteme numerice cu circuite integrate
I. Ristea ş.a. Manualul muncitorului electronist
M. Simionescu. Proiectare unitară a circuitelor electronice
C. Cluceru. Tehnica măsurărilor în telecomunicaţii
P. Niţulescu. Electroalimentarea instalaţiilor de telecomunicaţii
R. Răpeanu ş.a. Circuite integrate analogice. Catalog
Şt. Lozneanu ş.a. Casetofoane. Depanare. Funcţionare
T. Rădulescu ş.a. Centrale telefonice automate
N. Iosif ş.a. Tiristoare şi module de putere. Catalog
P. Postelnicu. Sisteme şi linii de transmisiuni telefonice
M. Silişteanu ş.a. Receptoare TV în culori
V. Baltac ş.a. Sisteme interactive şi limbaje conversaţionale
V. Baltac ş.a. Calculatoare electronice, grafica interactivă, prelucrarea imaginilor

Prof. dr. ing. **Adrian Petrescu**
– coordonator –
Prof. emerit **Gheorghe Rizescu**
Ing. asistent univ. **Francisc Iacob**
Ing. asistent univ. **Cornel Constantinescu**
Ing. **Tiberiu Ilin**
Ing. **Eduard Decsov**
Ing. **Constantin Novăcescu**
Ing. **Agota Matekovits**
Ing. **Florian Bar**
Ing. **Radu Berindeanu**
Elev **Iacob Petrescu**

Totul despre ... calculatorul personal aMIC

Volumul 2

Editura Tehnică
București, 1985

Colectivul de elaborare al cărţii cuprinde specialişti de la Institutul Politehnic Bucureşti, Liceul „Dimitrie Cantemir", Bucureşti, Întreprinderea de Memorii Electronice Timişoara, ITCI — Timişoara şi „Electrotimiş" Timişoara.

Contribuţia autorilor este următoarea :

A. Petrescu : coordonare, cap : 8 (p), 9 (p), 12 (p)
Gh. Rizescu : cap : 10 (p), 12 (p)
F. Iacob : cap : 8 (p), 12 (p)
C. Constantinescu : cap : 9 (p), 12 (p)
T. Ilin : cap : 8 (p), 11 (p), 13 (p)
E. Decsov : cap : 8 (p), 11 (p), 13 (p)
C. Novăcescu : cap : 8 (p), 14 (p)
A. Matekovits : cap : 10 (p), 11 (p), 12 (p), 13 (p), 14 (p)
F. Bar : cap : 9 (p), 14 (p)
R. Berindeanu : cap : 8 (p), 14 (p)
I. Petrescu : cap : 10 (p), 12 (p), anexa 3.

Recenzie : dr. ing. ADRIAN DAVIDOVICIU

Redactor : ing. PAUL ZAMFIRESCU

Culegerea şi paginarea realizată de o
echipă coordonată de EDUARD GIESSER

Coperta : **Arh. SILVIA MÎRŢU**
Desene : **LAURENŢIU ILIESCU**
Tehnoredactor : **ELLY GORUN**

Bun de tipar 10 dec. 1985 Coli tipar, 17,5
C. Z. 681.142

Tiparul a fost executat sub comanda nr. 110
la Întreprinderea Poligrafică „Banat"
Timişoara, Calea Aradului nr. 1.
Republică Socialistă România.

Cuprins*)

Vol. I

*) În volumul I, la paginile 13—20 se află cuprinsul extins al ambelor volume.

Vol. II

Capitolul 8.

Cuplări de echipamente periferice, interconectări şi aplicaţii ale microcalculatorului aMIC.

În acest capitol se vor prezenta o serie de aplicaţii, care urmăresc atît extinderea gamei de echipamente periferice, cît şi utilizarea microcalculatorului în cadrul unor sisteme dedicate. Astfel, se descriu aplicaţii referitoare la :

— cuplarea la microcalculatorul aMIC a unor LED-uri şi comutatoare.
— cuplarea la microcalculatorul aMIC a unui convertor numeric-analogic,
— cuplarea microcalculatorului aMIC cu microcalculatoarele FELIX M18, M118,
— cuplarea la calculatorul aMIC a unui echipament de desenare şi indicare pe ecran de tip „JOYSTICK",
— cuplarea unui convertor analog-numeric la microcalculatorul aMIC,
— utilizarea microcalculatorului aMIC pentru simularea unui circuit logic,
— cuplarea miniimprimantei MIM40 la microcalculatorul aMIC,
— cuplarea unui programator de memorii de tip EPROM la microcalculatorul aMIC,
— microcalculatorul aMIC cuplat cu terminalul DAF2010,
— interfaţarea microcalculatorului aMIC cu un minirobot,
— folosirea microcalculatorului aMIC în cadrul unui echipament pentru testarea microsistemelor orientate pe magistrală.
— utilizarea microcalculatorului aMIC ca unitate pentru deservirea maşinilor unelte,
— utilizarea microcalculatorului aMIC ca sistem de înregistrare/redare a parametrilor semicontinui de proces,
— utilizarea microcalculatorului aMIC în laborator pentru prelucrarea datelor provenite din analiza cromatografică.

8.1. Cuplarea unor LED-uri şi comutatoare

Utilizînd interfaţa paralelă de care dispune microcalculatorul personal se pot conecta la acesta diferite dispozitive numerice. Cel mai simplu exemplu îl reprezintă cuplarea unor led-uri şi comutatoare.

Interfaţa paralelă este constituită dintr-un circuit 8255 suplimentar, care are liniile porturilor A, B şi C, fiecare de cîte 8 biţi, legate la un conector de pe carcasa microcalculatorului. La portul A s-au conectat 8 comutatoare, fiecare avînd posibilitatea să furnizeze independent nivelele logice 1 sau 0.

La portul C s-au conectat 8 led-uri, care vor fi stinse sau aprinse, după cum biţii corespunzători ai circuitului 8255 sînt 1 sau Ø. Schema este prezentată în figura 8.1.

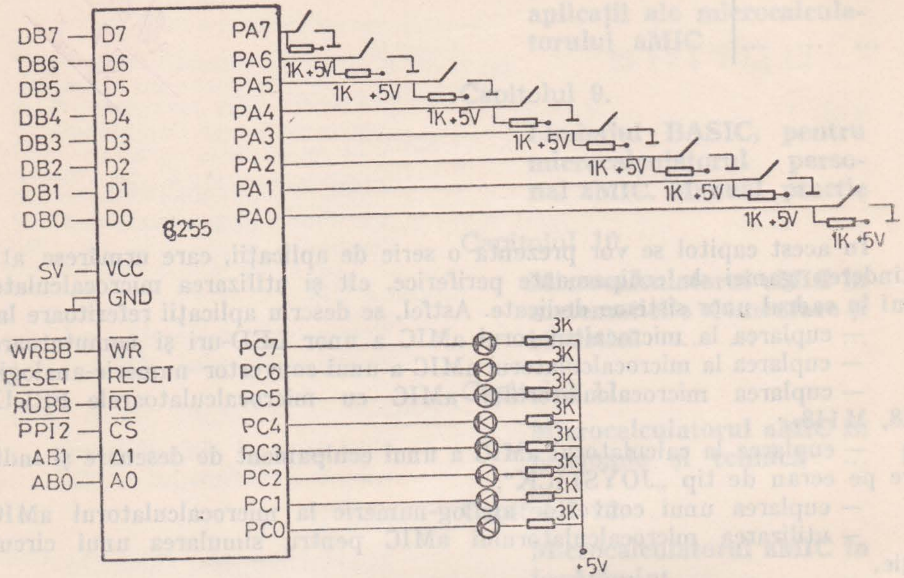


Fig. 8.1. Cuplarea unor led-uri şi comutatoare.

Pentru această aplicaţie circuitul 8255 suplimentar a fost programat în modul Ø de lucru cu portul C pentru ieşire, iar porturile A şi B pentru intrare. Adresele acestor porturi sînt :

— 4ØH : portul A ; — 42H : portul C ;
— 41H : portul B ; — 43H : portul de comandă.

În continuare se prezintă programul care realizează citirea permanentă a comutatoarelor şi afişarea stării acestora la led-uri.

Program de aplicaţie pentru interfaţa paralelă

```
START : MVI   A, 92H  ; încarcă în acumulator cuvîntul de comandă
        OUT   43H     ; transmite la circuitul 8255
BUCLA : IN    4ØH     ; citeşte portul A în acumulator
        CMA           ; complementează acumulatorul
        OUT   42H     ; transmite în portul C
        JMP   BUCLA   ; reluare.
```

8.2. Cuplarea unui convertor numeric-analogic

Microcalculatorul personal poate comanda un proces analogic simplu prin interfaţarea unui convertor numeric-analogic. Valoarea numerică ce este furnizată convertorului se obţine ca urmare a unor calcule sau prin prelucrarea unor date externe.

În acest exemplu s-a realizat interfaţarea unui convertor numeric-analogic pe 8 biţi, DACØ8, la magistrala sistemului. Acest lucru a fost posibil deoarece semnalele magistralei de date, adresele şi comenzile au fost legate la un conector de pe carcasă. Schema bloc a montajului este prezentată în figura 8.2.

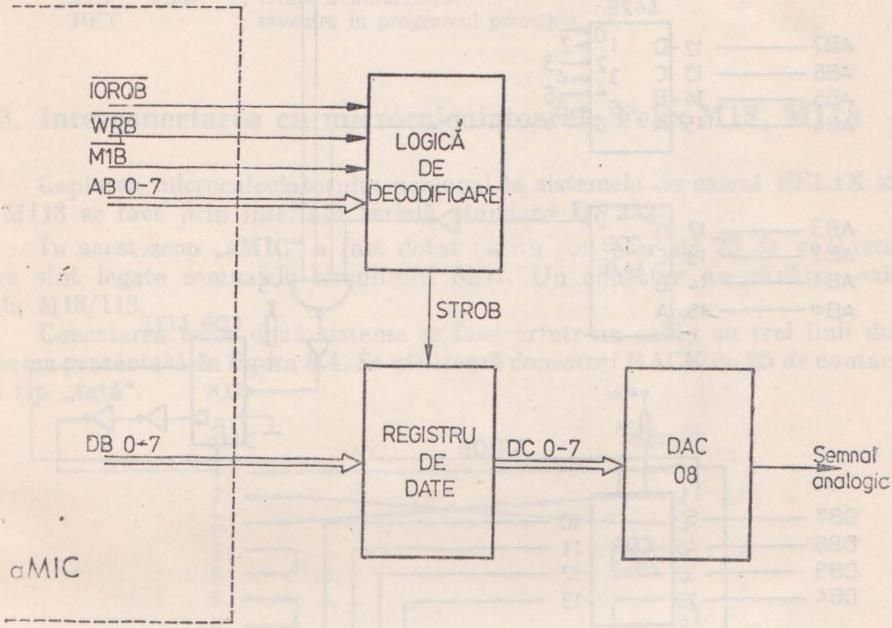


Fig. 8.2. Schema bloc pentru cuplarea unui convertor numeric analogic.

Aceasta se compune din logica de decodificare a adresei, un registru de date de 8 biţi şi convertorul DACØ8. Schema electrică completă a interfaţării convertorului se prezintă în figura 8.3.

Realizarea semnalului analogic de la ieşire constă în înscrierea în registrul de date a unor valori numerice pe 8 biţi la momente de timp bine stabilite. În continuare se prezintă programul în limbaj de asamblare care realizează la ieşirea convertorului un semnal în dinţi de fierăstrău. Transmiterea datelor se face prin instrucţiuni de ieşire (OUT) cu adresa 6ØH.

Program de aplicaţii pentru DACØ8.

```
START : MVI   A, Ø    ; iniţializarea acumulatorului cu zero
BUCLA : OUT   6ØH     ; ieşire în registrul de date
        INR   A       ; incrementează acumulatorul
        JMP   BUCLA   ; salt la reluare.
```

Pentru a varia perioada semnalului analogic se poate utiliza o subrutină de aşteptare;

```
START : MVI   A, Ø    ; iniţializarea acumulatorului cu zero
BUCLA : OUT   6ØH     ; înscrie în registrul de date
        INR   A       ; incrementează acumulatorul
        CALL  WAIT    ; reglează perioada semnalului
        JMP   BUCLA   ; salt la reluare
```

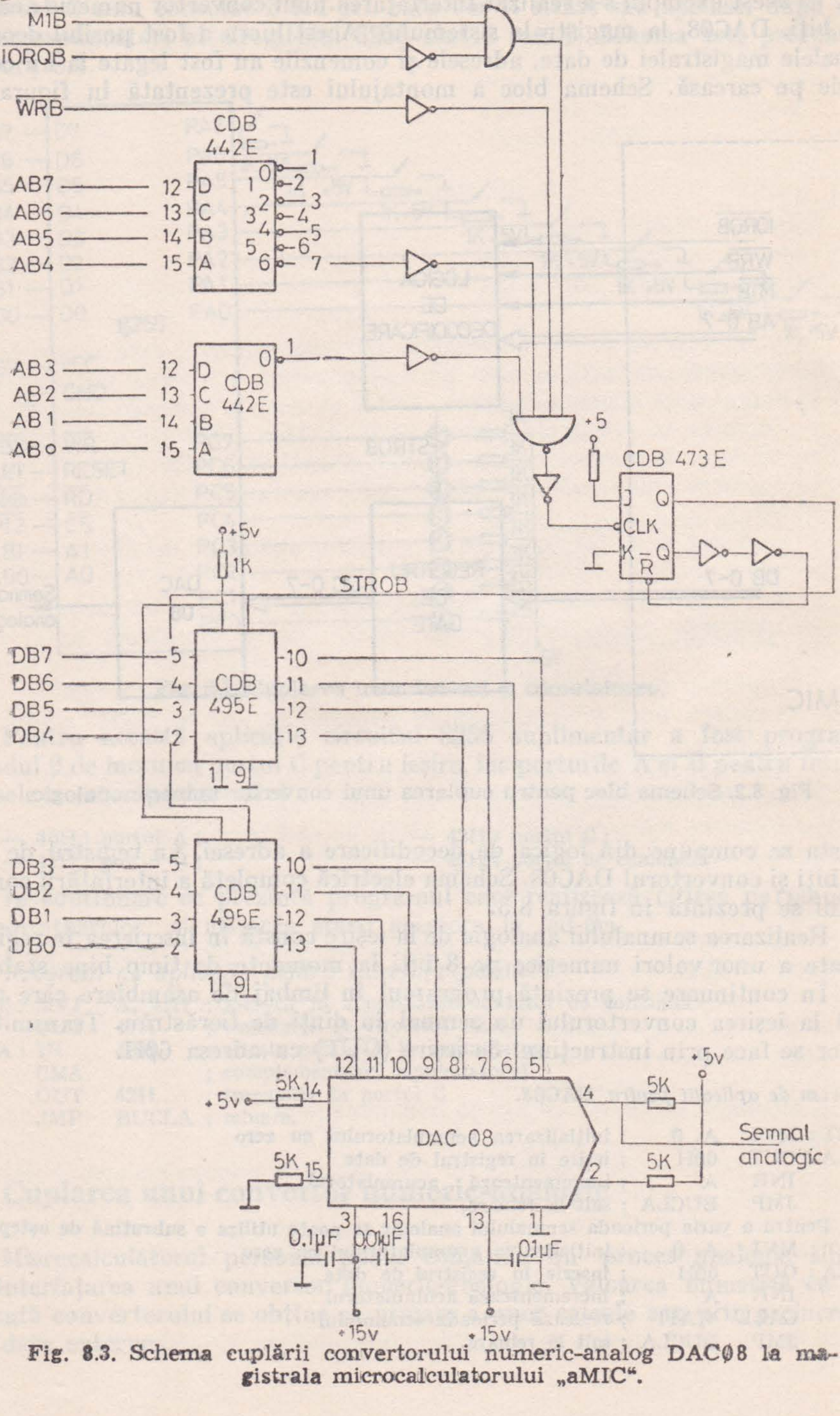


Fig. 8.3. Schema cuplării convertorului numeric-analog DAC08 la magistrala microcalculatorului „aMIC".

```
WAIT : LXI   D, 5ØØH ; această valoare reglează perioada
       PUSH  PSW     ; salvează acumulatorul
ET1 :  DCX   D       ; decrementează contorul
       MOV   A, D    ; încarcă registrul D în A
       ORA   E       ; SAU logic între A și E
       JNZ   ET1     ; reluare dacă diferit de zero
       POP   PSW     ; reface acumulatorul
       RET           ; revenire în programul principal.
```

8.3. Interconectarea cu microcalculatoarele Felix M18, M118

Cuplarea microcalculatorului personal la sistemele de calcul FELIX M18 și M118 se face prin interfața serială standard RS 232.

În acest scop „aMIC" a fost dotat cu un con ctor de 25 de contacte la care sînt legate semnalele circuitului 8251. Un conector asemănător există și la M18/118.

Conectarea celor două sisteme se face printr-un cablu cu trei linii după schema prezentată în figura 8.4. Se utilizează conectori RACK cu 25 de contacte, de tip „tată".

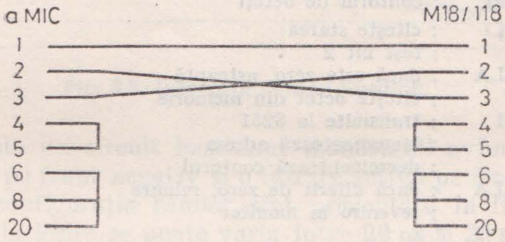


Fig. 8.4. Cablul de interconectare „aMIC" — FELIX M18/118.

Ceilalți pini din conectori, neindicați în desen, sînt lăsați neconectați.

Pentru realizarea comunicației pe fiecare din cele două sisteme rulează cîte un program. În continuare se prezintă un exemplu care transferă 1ØØ de octeți din memoria microcalculatorului FELIX M18/118 de la adresa 6ØØØH în memoria microcalculatorului „aMIC" la adresa 66ØØH. Un asemenea procedeu este deosebit de util, deoarece pe M18/118 se pot introduce și pune la punct programe complexe pentru „aMIC" utilizîndu-se resurse software evoluate disponibile pe aceste sisteme, apoi aceste programe sînt transferate la microcalculatorul personal pentru a fi rulate.

În scopul realizării transferului de date pe microcalculatorul „aMIC" se rulează un program de recepție a octeților transmiși pe linia serială.

```
; Program de recepție a datelor pe linia serială

START : MVI  A, ØCEH ; cuvîntul de mod
        OUT  1       ; transmite la 8251
        MVI  A, 27H  ; cuvîntul de comandă
```

```
         OUT   1        ; transmite la 8251
         LXI   H, 66ØØH ; în H, L adresa de început
         MVI   C, 1ØØ   ; contorul de octeţi
BUCLA :  IN    1        ; bucla de citire a stării circuitului 8251
         ANI   2        ; test bit 1
         JZ    BUCLA    ; dacă Ø, aşteaptă
         IN    Ø        ; citeşte octet
         MOV   M, A     ; înscrie în memorie
         INX   H        ; incrementează adresa
         DCR   C        ; decrementează contorul
         JNZ   BUCLA    ; dacă diferit de zero, reface
         JMP   Ø        ; altfel intră în monitor
```

Pe microcalculatorul FELIX M18/118 rulează un program de transmisie a octeţilor din memorie pe linia serială.

Program de transmisie a datelor pe linia serială

```
START :  MVI   A, 4ØH   ; cuvîntul de comandă pentru 8251
         OUT   ØF7H     ; realizează iniţializarea circuitului
         MVI   A, ØCEH  ; cuvîntul de mod
         OUT   ØF7H     ; transmite la 8251
         MVI   A, 27H   ; cuvîntul de comandă
         OUT   ØF7H     ; transmite la 8251
         LXI   H, 6ØØØH ; în H, L adresa de început
         MVI   C, 1ØØ   ; contorul de octeţi
BUCLA :  IN    ØF7H     ; citeşte starea
         ANI   4        ; test bit 2
         JZ    BUCLA    ; dacă este zero, aşteaptă
         MOV   A, M     ; citeşte octet din memorie
         OUT   ØF6H     ; transmite la 8251
         INX   H        ; incrementează adresa
         DCR   C        ; decrementează contorul
         JNZ   BUCLA    ; dacă diferit de zero, reluare
         RST   Ø        ; revenire în monitor
```

Cele două programe se execută în acelaşi timp, dar cu menţiunea că programul de recepţie a datelor se lansează primul.

8.4. Cuplarea unui JOYSTICK

Joystick-ul este un dispozitiv utilizat în conjuncţie cu terminalele grafice pentru specificarea manuală, de către utilizator, a coordonatelor unui punct pe ecran.

Joystick-ul constă dintr-o articulaţie sferică, cu o manetă fixată de partea sferică mobilă, ale cărei deplasări unghiulare sînt transformate în variaţii ale rezistenţei electrice a doi potenţiometri, corespunzînd coordonatei orizontale, respectiv verticale. Variaţiile de rezistenţă sînt transformate apoi în variaţii de semnal electric, care — la rîndul lor — sînt convertite în valori numerice. Aceste valori numerice sînt preluate de sistemul de calcul şi, prin metode software sau hardware (la echipamentele care cer viteză mare de lucru), sînt transformate într-un punct sau cursor grafic, afişat pe ecran. Este posibilă afişarea întregului traseu impus punctului curent, adică desenarea comandată

de la joystick. Este, de asemenea, posibilă introducerea unor funcții suplimentare, care să permită utilizarea acestui dispozitiv în diferite aplicații.

Cuplarea unui joystick la microcalculatorul „aMIC" se face prin intermediul interfeței paralele, utilizînd numai patru biți ai circuitului 8255. Schema prezentată în figura 8.5 conține două circuite CDB4121E.

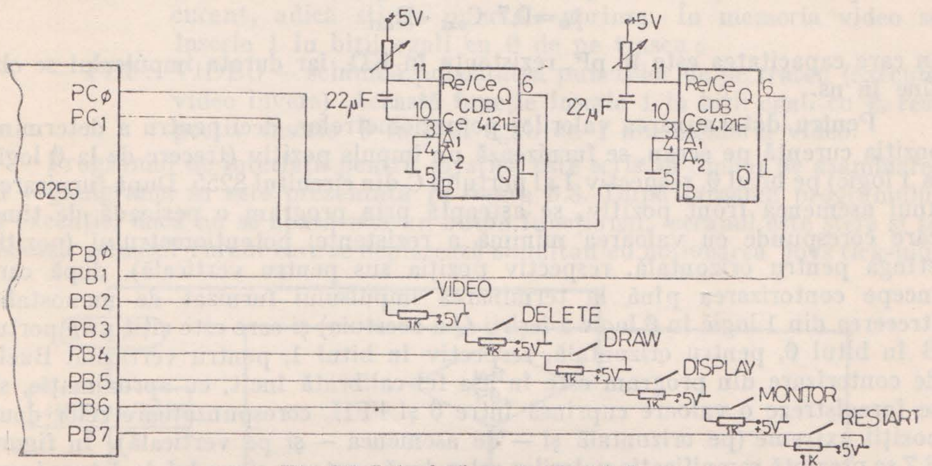
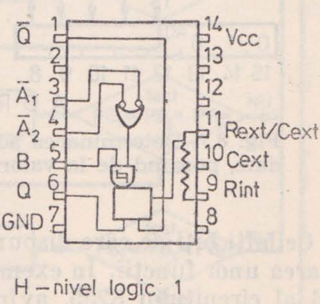


Fig. 8.5. Interfață pentru joystick.

CDB4121E este un circuit basculant monostabil avînd două intrări de condiționare active pe front negativ și o intrare activă pe front pozitiv. Tabela de funcționare și configurația pinilor sînt prezentate în figura 8.6. Durata impulsului obținut la ieșire se poate varia între 2∅ ns și 28 s utilizînd componente pasive externe. Fără componente externe, avînd intrarea Rint conectată

INTRĂRI			IEȘIRI	
$\bar{A}_1$	$\bar{A}_2$	B	Q	$\bar{Q}$
L	X	H	L	H
X	L	H	L	H
X	X	L	L	H
H	H	X	L	H
H	↓	H	⊓	⊔
↓	H	H	⊓	⊔
↓	↓	H	⊓	⊔
L	X	↑	⊓	⊔
X	L	↑	⊓	⊔



H – nivel logic 1
L – nivel logic 0

Fig. 8.6. Tabela de funcționare și configurația pinilor la circuitul CDB 421E.

la Vcc, iar intrările C_{ext} și R_{ext}/C_{ext} lăsate neconectate, se obține un impuls cu durata specifică de 3∅—35 ns.

Pentru valori ale capacității externe cuprinse între 1∅ pF și 1∅ μF iar ale rezistenței externe cuprinse între 2 KΩ și 4∅ KΩ durata impulsului t se calculează cu formula :

$$t_w = 0{,}7\ C_{ext} \cdot R_{ext}$$

în care capacitatea este în pF, rezistența în KΩ, iar durata impulsului se obține în ns.

Pentru determinarea valorilor potențiometrelor, deci pentru a determina poziția curentă pe ecran, se furnizează un impuls pozitiv (trecere de la ∅ logic la 1 logic) pe bitul ∅, respectiv 1 al portului C din circuitul 8255. După furnizarea unui asemenea front pozitiv, se așteaptă prin program o perioadă de timp care corespunde cu valoarea minimă a rezistenței potențiometrului (poziția stînga pentru orizontală, respectiv poziția sus pentru verticală), după care începe contorizarea pînă la terminarea impulsului furnizat de monostabil (trecerea din 1 logic în ∅ logic a ieșirii Q a acestuia) și care este citit prin portul B în bitul ∅, pentru orizontală, respectiv în bitul 1, pentru verticală. Bucla de contorizare din program este în așa fel calibrată încît, cu aproximație, să se înregistreze o valoare cuprinsă între ∅ și FFH, corespunzătoare celor două poziții extreme (pe orizontală și — de asemenea — și pe verticală). În figura 8.7 se prezintă semnificația valorilor celor două contoare și modul de determinare a adresei octetului pentru memoria ecran și al bitului din cadrul octetului de date. Deoarece memoria ecran este cuprinsă între adresele 4∅∅∅H—5FFFH, în pozițiile cele mai semnificative A15—A13 se găsește configurația ∅1∅.

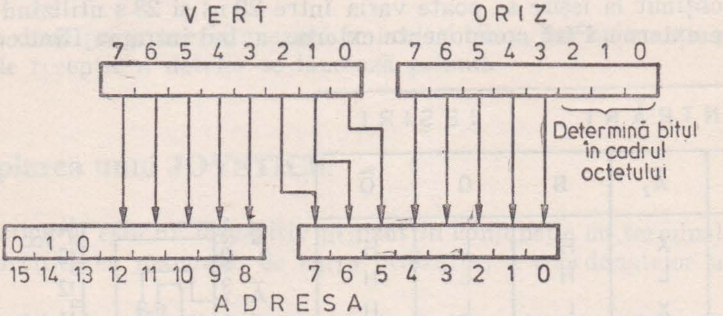


Fig. 8.7. Determinarea adresei din memoria video și octetul de date, pornind de la valorile celor două contoare VERT și ORIZ.

Ceilalți biți de care dispune interfața paralelă se pot utiliza pentru implementarea unor funcții. În exemplul considerat se utilizează încă 6 biți din portul B al circuitului 8255, avînd următoarele semnificații :

— PB7 : RESTART — relansează programul de joystick ;

— PB6 : MONITOR — execută revenirea în monitor (start de la adresa ∅)

— PB5 : DISPLAY — anulează o funcție anterioară, realizînd numai afișarea punctului curent, fără să modifice imaginea de pe ecran (deci nici memoria video) ;

— PB4 : DRAW — marchează traseul parcurs de la punctul curent pe ecran prin puncte aprinse (biți 0 în memoria video) :

— PB3 : DELETE — șterge contururile de pe traseul parcurs de punctul curent, adică stinge punctele aprinse. În memoria video se înscrie 1 în biții egali cu ∅ de pe traseu ;

— PB2 : VIDEO — schimbă polaritatea punctelor de pe traseu (execută video invers). Această funcție înscrie 1 în biții egali cu ∅, respectiv înscrie ∅ în biții egali cu 1 din memoria video.

Programul de interfață pentru joystick este scris în limbaj de asamblare, iar organigrama sa este prezentată în figura 8.8. După lansarea programului în execuție, dacă nu se apasă nici un buton funcțional, ecranul este șters și se afișează punctul curent care se deplasează simultan cu acționarea joystick-ului.

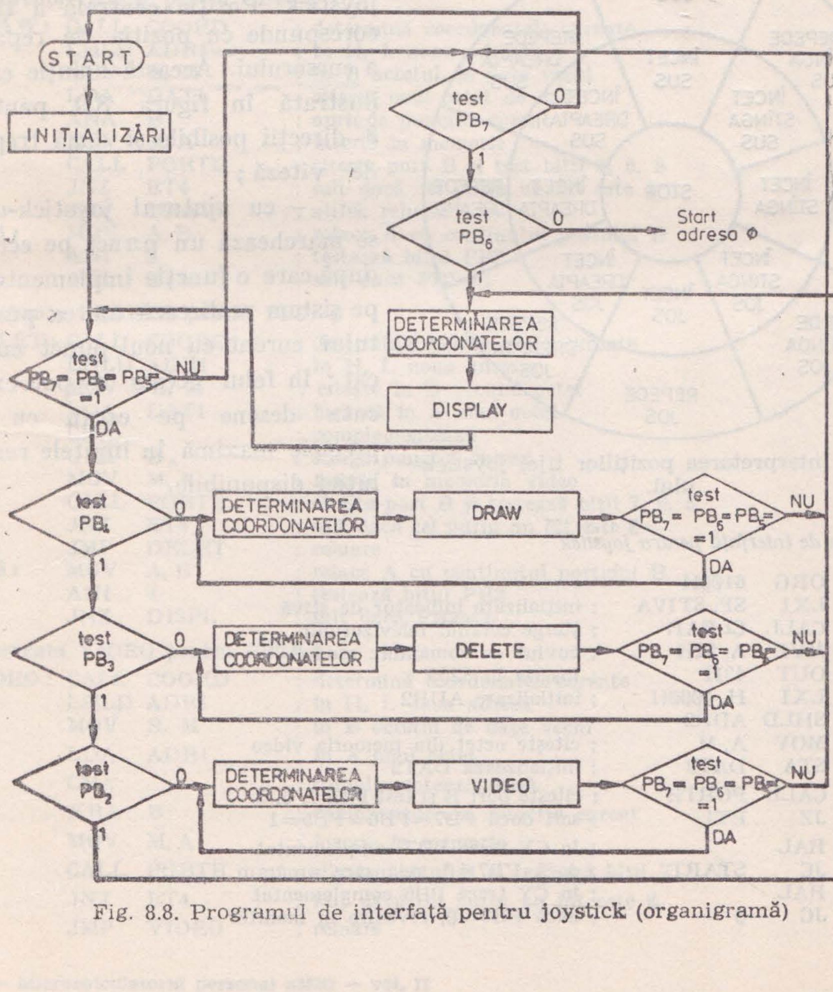


Fig. 8.8. Programul de interfață pentru joystick (organigramă)

Dacă se apasă oricare din butoanele DRAW, DELETE sau VIDEO, se execută în continuare funcţia respectivă (fără ca să se menţină butonul apăsat), pînă cînd se apasă unul din butoanele RESTART, MONITOR sau DISPLAY.

Programul prezentat în acest exemplu realizează o traducere a poziţiei articulaţiei joystick-ului în coordonate pe ecran (respectiv adresă şi octet de date pentru memoria video). Dezavantajul acestei soluţii este o sensibilitate deosebită a punctului curent (cursorului) de pe ecran faţă de orice mişcări ale tijei dispozitivului. Pentru eliminarea acestui neajuns se pot utiliza alte metode de interfaţare a joystick-ului, ca de exemplu:

— poziţia tijei (a articulaţiei) să indice direcţia dintr-un număr finit şi eventual mic de direcţii posibile şi viteza (trepte discrete de viteză) pentru cursorul de pe ecran. Deplasarea se face din poziţia curentă în direcţia şi cu viteza date de joystick. Poziţia centrală a tijei corespunde cu poziţia de repaus a cursorului. Această soluţie este ilustrată în figura 8.9, pentru 8 direcţii posibile şi două trepte de viteză;

— cu ajutorul joystick-ului se marchează un punct pe ecran după care o funcţie implementată pe sistem realizează unirea punctului curent cu noul punct marcat; în felul acesta se pot executa desene pe ecran cu o precizie maximă în limitele rezoluţiei disponibile.

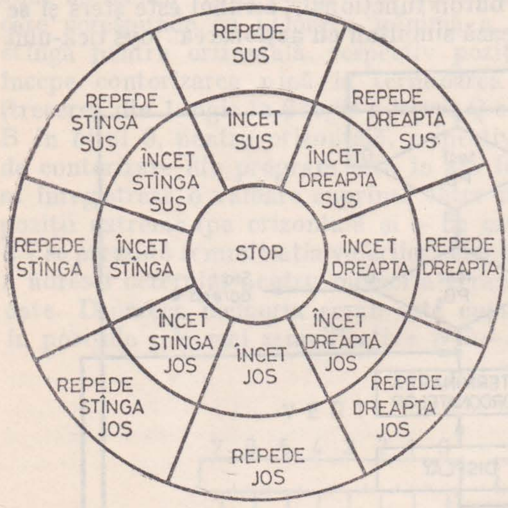


Fig. 8.9. Interpretarea poziţiilor tijei joystick-ului.

```
; Program de interfaţă pentru joystick

        ORG   61ØØH
START : LXI   SP, STIVA    ; iniţializare indicator de stivă
        CALL  CLEAR        ; şterge ecranul televizorului
        MVI   A, 92H       ; cuvînt de comandă : mod Ø
        OUT   43H          ; înscrie în 8255
        LXI   H, 4ØØØH     ; iniţializare ADR2
        SHLD  ADR2
        MOV   A, M         ; citeşte octet din memoria video
        STA   DAT2         ; iniţializează DAT2
ET2 :   CALL  PORTB        ; citeşte port B şi test biţii 7, 6, 5
        JZ    ET1          ; salt dacă PB7=PB6=PB5=1
ET4 :   RAL                ; în CY trece PB7 complementat
        JC    START        ; dacă PB7=Ø, relansare program
        RAL                ; în CY trece PB6 complementat
        JC    Ø            ; dacă PB6=Ø, revenire în monitor
```

```
; secvenţa DISPLAY pentru PB5=∅
DISPL : CALL  COORD   ; determină coordonatele curente
        LHLD  ADR2    ; în H, L vechea adresă
        LDA   DAT2    ; în H, L vechiul octet
        MOV   M, A    ; reface în memoria video
        LHLD  ADR1    ; în H, L noua adresă
        MOV   A, M    ; citeşte octetul de date
        STA   DAT2    ; salvează data
        SHLD  ADR2    ; şi adresa
        MOV   B, A    ; salvează în registrul B
        LDA   DAT1    ; încarcă în A noul octet de date
        ANA   B       ; aprinde punctul curent
        MOV   M, A    ; înscrie în memoria video
        JMP   ET2     ; reluare
ET1 :   MOV   A, B    ; reface A cu conţinutul portului B
        ANI   1∅H     ; testează bitul PB4
        JNZ   ET3     ; salt dată PB4=1
; secvenţa DRAW pentru PB4=∅
DRAW :  CALL  COORD   ; determină coordonatele curente
        LHLD  ADR1    ; în H, L noua adresă
        MOV   B, M    ; în B octetul de date vechi
        LDA   DAT1    ; citeşte noul octet de date
        ANA   B       ; aprinde punctul curent
        MOV   M, A    ; înscrie în memorie
        CALL  PORTB   ; citeşte port B şi test biţii 7, 6, 5
        JNZ   ET4     ; salt dacă cel puţin un bit este ∅
        JMP   DRAW    ; altfel, reluare
ET3 :   MOV   A, B    ; reface A cu conţinutul portului B
        ANI   8       ; testează bitul PB3
        JNZ   ET5     ; salt dacă PB3=1
; secvenţa DELETE pentru PB3=∅
DELET : CALL  COORD   ; determină coordonatele curente
        LHLD  ADR1    ; în H, L noua adresă
        MOV   B, M    ; citeşte în B vechiul octet
        LDA   DAT1    ; încarcă în A noul octet
        CMA           ; complementează
        ORA   B       ; stinge punctul curent
        MOV   M, A    ; înscrie în memoria video
        CALL  PORTB   ; citeşte port B şi testează biţii 7, 6, 5
        JNZ   ET4     ; salt dacă cel puţin un bit este ∅
        JMP   DELET   ; reluare
ET5 :   MOV   A, B    ; reface A cu conţinutul portului B
        ANI   4       ; testează bitul PB2
        JNZ   DISPL   ; salt dacă PB2=1
; secvenţa VIDEO pentru PB2=∅
VIDEO : CALL  COORD   ; determină coordonatele curente
        LHLD  ADR1    ; în H, L noua adresă
        MOV   B, M    ; în B octetul de date vechi
        LDA   ADR1    ; în A noul octet
        CMA           ; complementează octet
        XRA   B       ; complementează punctul curent
        MOV   M, A    ; înscrie în memorie
        CALL  PORTB   ; citeşte portul B şi testează biţii 7, 6, 5
        JNZ   ET4     ; salt dacă cel puţin un bit este ∅
        JMP   VIDEO   ; reluare
```

```
; subrutina COORD
; determină coordonatele punctului curent
; furnizează la ADR1 adresa din memoria video
; şi la DAT1 octetul de date
COORD : MVI  D, 2     ; setează bitul 1
        CALL XY       ; determină coordonata X
        STA  ORIZ     ; memorează şi variabila ORIZ
        MVI  D, 1     ; setează bitul Ø
        CALL XY       ; determină coordonata Y
        STA  VERT     ; memorează în variabila VERT
; secvenţa care determină adresa pentru memoria ecran
; şi octetul de date
        LXI  H, 4ØØØH ; iniţializare H, L
        LDA  ORIZ     ; citeşte contorul pentru orizontală
        ANI  ØFBH     ; anulează biţii A2-AØ
        RRC           ; aduce A7-A3 în poziţiile A4-AØ
        RRC
        RRC
        MOV  L, A     ; încarcă în registrul L
        LDA  VERT     ; citeşte contorul pentru verticală
        ANI  7        ; preia biţii A2-AØ
        RRC           ; aduce informaţia în poziţiile A7-A5
        RRC
        RRC
        ORA  L        ; concatenează cu informaţia din L
        MOV  L, A     ; încarcă în registrul L octetul mai puţin semnificativ de
                      ; adresă
        LDA  VERT     ; citeşte contor pentru verticală
        ANI  ØF8H     ; preia biţii A7-A3
        RRC           ; aduce în poziţiile A7-AØ
        RRC
        RRC
        ORA  H        ; concatenează cu informaţia din H
        MOV  H, A     ; încarcă în registrul H octetul mai semnificativ de adresă
        LHLD ADR1     ; depunde adresa la ADR1
        LDA  ORIZ     ; citeşte contorul pentru orizontală
        ANI  7        ; preia biţii A2-AØ
        MOV  L, A     ; încarcă în L
        MVI  A, 7FH   ; iniţializare octet date
        INR  L
AD2 :   DCR  L        ; poziţionează indicatorii
        JZ   AD1      ; dacă L=Ø, stop
        RRC           ; altfel, deplasează bit=Ø (punct aprins)
        JMP  AD2      ; reluare
AD1 :   STA  DAT1     ; depune octetul de date la adresa DAT1
        RET
; subrutina XY
; poziţionează contorul pentru orizontală
; sau pentru verticală
; furnizează rezultatul în A (contor între Ø şi FFH)
XY :    MVI  A, Ø     ; bit 0, bit 1=0 din portul C al circuitului 8255
        OUT  42H      ; trimite la 8255
        MOV  A, D     ; poziţionează pe 1 bitul corespunzător pentru orizontală
                      ; sau pentru verticală
        OUT  42H      ; trimite la 8255, declanşează monostabilul
        MVI  C, ØFFH  ; iniţializare contor
XY1 :   IN   41H      ; citeşte port B al circuitului 8255
        ANA  D        ; test bit 1/Ø
```

```
        JZ    XY1      ; aşteaptă impulsul monostabilului
        MVI   A, 4     ; secvenţa se execută de patru ori
WAIT:   MVI   B, 84H   ; contor
        DCR   B
        JNZ   WAIT+2
        DCR   A
        JNZ   WAIT
; începe incrementarea contorului
XY2:    INR   C
        IN    41H      ; citeşte port B
        ANA   D        ; test bit Ø/1
        NOP            ; calibrarea contorizării
        NOP
        NOP
        JNZ   XY2      ; aşteaptă terminarea impulsului
        MOV   A, C     ; transferă rezultatul în A
        RET
; zona de variabile program
ORIZ:   DS    1        ; contorul pentru orizontală
VERT:   DS    1        ; contorul pentru verticală
ADR1:   DS    2        ; adresa nouă pentru memoria video
ADR2:   DS    2        ; adresa veche
DAT1:   DS    1        ; noul octet de date
DAT2:   DS    1        ; vechiul octet de date
        DS    1Ø       ; stiva program
STIVA:  DS    1
        END   START
```

8.5. Cuplarea unui conventor analog-numeric

Microcalculatorul personal are posibilitatea să efectueze măsurări de mărimi analogice, eventual extrase dintr-un proces, prin conectarea unui convertor analog-numeric. Interfaţarea convertorului la sistem se face prin circuitul 8255 suplimentar (interfaţa paralelă). Se poate executa conversia unei singure mărimi analogice sau a mai multora succesiv ; în acest ultim caz, mărimile analogice sînt aduse la intrarea convertorului prin intermediul unui multiplexor analogic. Selecţia intrărilor multiplexorului se face prin intermediul circuitului 8255.

Schema de principiu este prezentată în figura 8.1Ø. Se utilizează un multiplexor analogic 16 : 1, iar selecţia se realizează prin biţii PC3-PCØ ai portului C din circuitul 8255. Convertorul analog-numeric este pe 12 biţi, avînd o intrare de comandă START conversie, conectată la bitul PC7 al circuitului 8255. Un front pozitiv pe această intrare lansează operaţia de conversie. Biţii de date inferiori DØ—D7 sînt conectaţi la pinii PAØ-PA7, iar biţii de date superiori D8-D11 la pinii PBØ-PB3. De asemenea, convertorul mai dispune de un semnal pentru a anunţa sfîrşitul conversiei, STOP conversie, conectat la pinul PB7.

În continuare se prezintă lista instrucţiunilor de intrare/ieşire şi semnificaţiile fiecăreia :

OUT 42H ; biţii A3—AØ selectează una din cele 16 intrări analogice
OUT 42H ; bitul A7=1 lansează operaţia de conversie

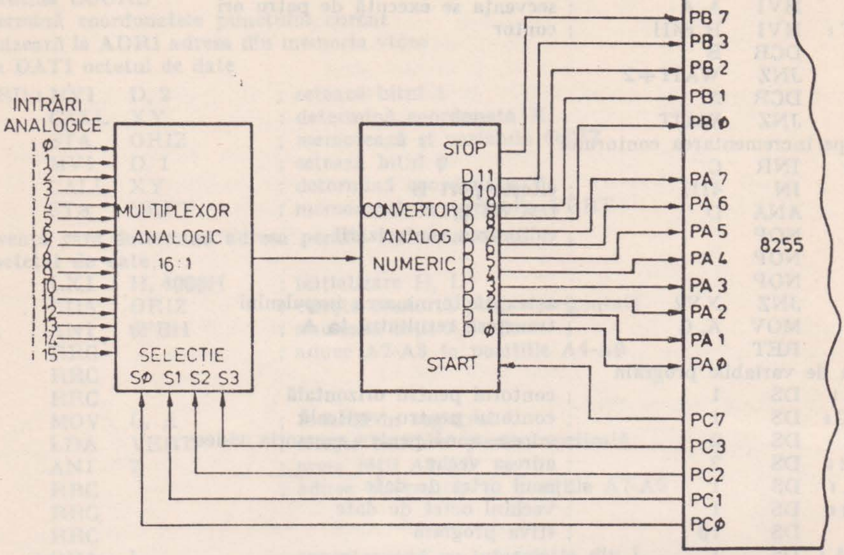


Fig. 8.10. Cuplarea unui convertor analog-numeric.

```
IN  41H ; citește în A7 bitul STOP conversie
IN  4ØH ; citește în A7—AØ biții de date D7—DØ numai după terminarea
        ; conversiei
IN  41H ; citește în A3—AØ biții de date D11—D8
OUT 42H ; bitul A7=Ø resetează convertorul
```

Programul care urmează execută succesiv conversia analog-numerică a celor 16 intrări înscriind rezultatele în memorie, fiecare pe cîte doi octeți începînd de la adresa LISTA.

```
; Program de interfațare a unui convertor analog-numeric

       MVI  A, 92H     ; programare circuit 8255
       OUT  43H        ; trimite în portul de comandă
       LXI  H, LISTA   ; H, L conțin adresa în zona de memorie
       MVI  B, Ø       ; selecție intrare analogică
ET1 :  MOV  A, B       ; transferă cod de selecție în A
       OUT  42H        ; resetează convertorul A7-Ø
       ORI  8ØH        ; A7=1
       OUT  42H        ; start conversie
ET2 :  IN   41H        ; citește starea
       RAL             ; A7=STOP trece în CY
       JNC  ET2        ; dacă STOP=Ø, așteaptă
       IN   4ØH        ; citește biții de date D7-DØ
       MOV  M, A       ; înscrie în memorie
       INX  H          ; incrementează adresa de memorie
       IN   41H        ; citește biții de date D11-D8
       ANI  ØFH        ; selectează biții A3-AØ
       MOV  M, A       ; înscrie în memorie
```

```
INX  H     ; incrementează adresa de memorie
INR  B     ; incrementează contorul de selecție
ANI  ØFH   ; test sfîrșit achiziție
JNZ  ET1   ; reia dacă nu este gata
JMP  NEXT  ; altfel execută prelucrarea datelor
```

8.6. Simularea unui circuit logic

Microcalculatorul „aMIC" poate fi utilizat în proiectarea schemelor logice, înlocuind faza de sinteză a circuitelor și de punere la punct cu ajutorul osciloscopului, analizorului logic etc.

În acest fel se poate economisi timp și efort material. Pentru exemplificare, se consideră o interfață simplificată care conectează un echipament periferic de ieșire la un microsistem construit tot cu microprocesorul Z8Ø (figura 8.11).

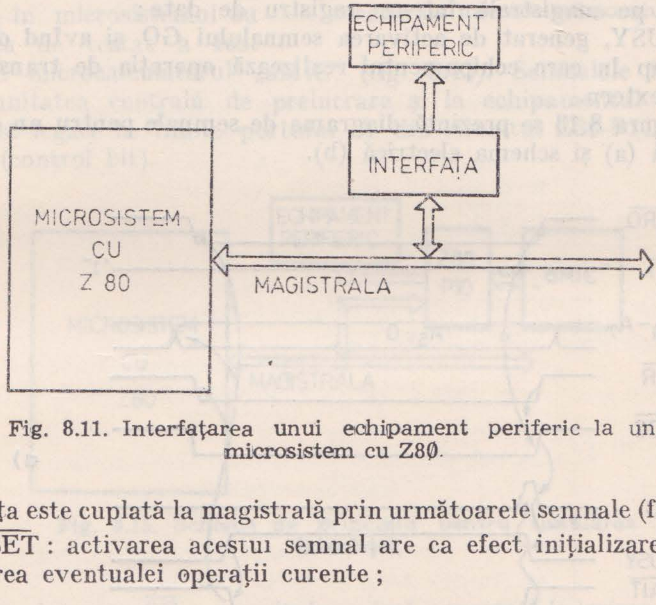


Fig. 8.11. Interfațarea unui echipament periferic la un microsistem cu Z8Ø.

Interfața este cuplată la magistrală prin următoarele semnale (figura 8.12) :

— $\overline{RESET}$: activarea acestui semnal are ca efect inițializarea interfeței și abandonarea eventualei operații curente ;

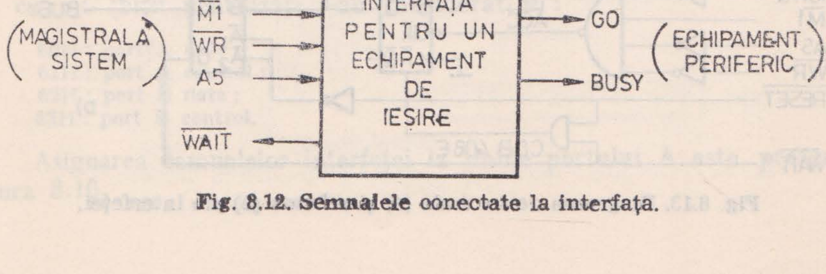


Fig. 8.12. Semnalele conectate la interfața.

— $\overline{IORQ}$ activ (∅ logic) şi $\overline{M1}$ inactiv (1 logic) indică ciclu de I/E ;

— $\overline{WR}$ activ (∅ logic) indică sensul transferului (de la UCP la echipamentul periferic).

— A5 este utilizat pentru selectarea interfeţei. Deoarece microsistemul are un număr redus de dispozitive de intrare/ieşire, în cadrul octetului de adresă de I/E biţii A7—A2 sînt utilizaţi pentru selectarea cîte unui circuit (fiecare bit, activ pe ∅, controlează un dispozitiv), iar biţii A1 şi A∅, eventual pentru selectarea porturilor interne. Deci în octetul de adresă pentru interfaţă bitul A5 este ∅, toţi ceilalţi biţi fiind 1.

— $\overline{WAIT}$ este conectat la intrarea $\overline{WAIT}$ a microprocesorului, permiţînd adaptarea vitezei de lucru a unităţii centrale de prelucrare la viteza echipamentului periferic.

Semnalele generate de interfaţă pentru echipamentul periferic de ieşire sînt următoarele :

— GO, generat la selectarea interfeţei şi utilizat pentru preluarea datelor de pe magistrală într-un registru de date ;

— BUSY, generat de activarea semnalului GO şi avînd o durată fixă, ∅,5 ms, timp în care echipamentul realizează operaţia de transfer a datelor pe suport extern.

În figura 8.13 se prezintă diagrama de semnale pentru un ciclu de acces la interfaţă (a) şi schema electrică (b).

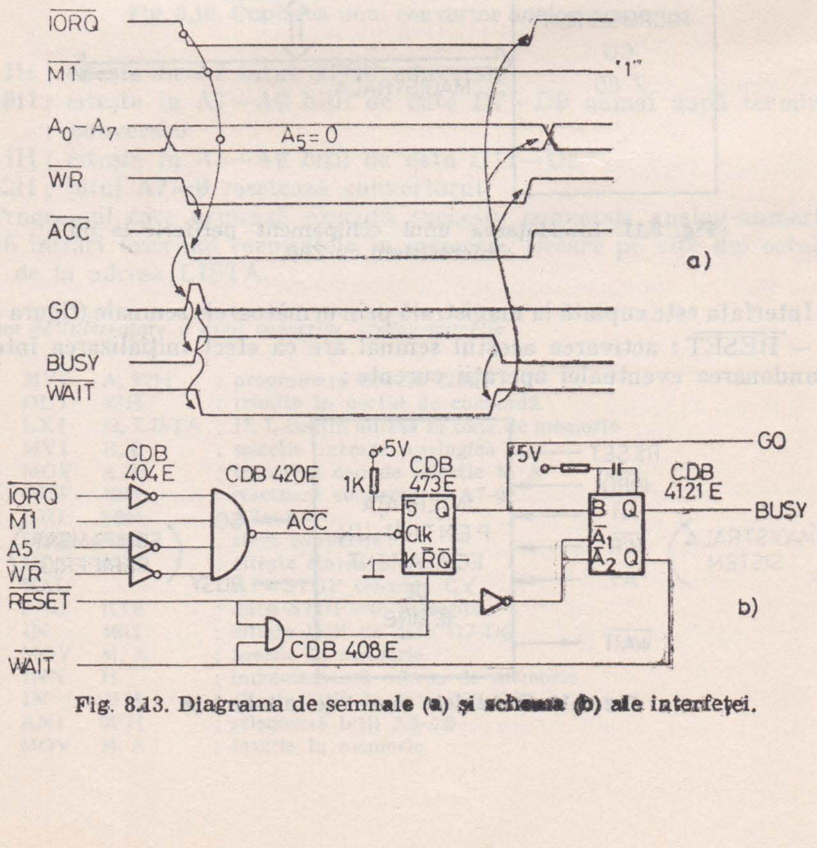


Fig. 8.13. Diagrama de semnale (a) şi schema (b) ale interfeţei.

În loc să se construiască fizic această schemă și să se verifice prin metode hardware, utilizînd osciloscop, analizor logic, etc. ... funcționarea corectă se poate verifica prin program, utilizînd microcalculatorul „aMIC". Conectarea microcalculatorului la microsistemul cu Z8∅ se poate face prin interfața paralelă. În acest scop se utilizează un circuit Z8∅–PIO cuplat extern la magistrala „aMIC"-ului. În figura 8.14 este prezentat circuitul Z8∅-PIO și semnalele de cuplare la magistrală.

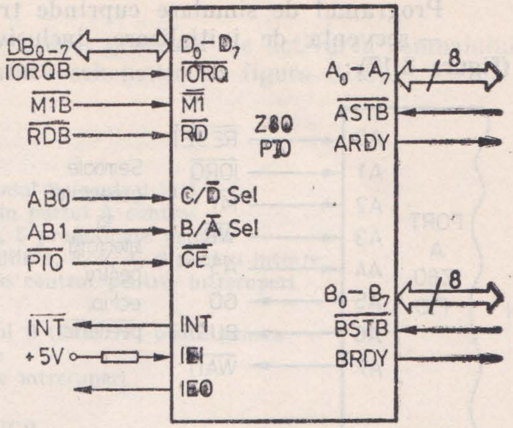


Fig. 8.14. Cuplarea circuitului Z8∅-PIO la magistrala microcalculatorului „aMIC".

Astfel, în microsistemul cu Z8∅ interfața de testat a fost înlocuită cu microcalculatorul „aMIC" (fig. 8.15). Semnalele interfeței de cuplare la unitatea centrală de prelucrare și la echipamentul periferic de ieșire, au fost legate la liniile portului A din circuitul Z8∅-PIO, programat în modul 3 (control bit).

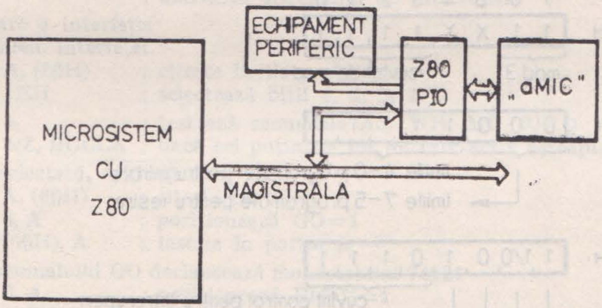


Fig. 8.15. Schema de principiu pentru simularea interfeței.

Deoarece intrarea $\overline{CE}$ a circuitului a fost conectată la ieșirea 3 a decodificatorului (CDB442E) de adrese de I/E (semnalul $\overline{PIO}$), iar terminalele $C/\bar{D}$ Sel și $B/\bar{A}$ Sel la liniile de adresă AB∅, respectiv AB1 rezultă adresele porturilor din circuit (biții neutilizați s-au considerat ∅):

6∅H : port A date ;
61H : port A control ;
62H : port B date ;
63H : port B control.

Asignarea semnalelor interfeței la liniile portului A este prezentată în figura 8.16.

Programul de simulare cuprinde trei secvențe principale:

— secvența de inițializare, inclusiv programarea circuitului Z8Ø-PIO (figura 8.17);

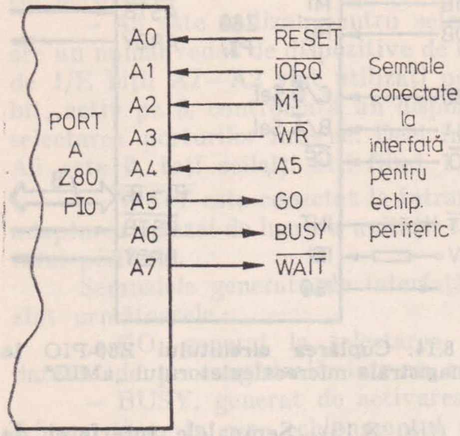


Fig. 8.16. Asignarea semnalelor interfeței la liniile portului A din Z8Ø-PIO.

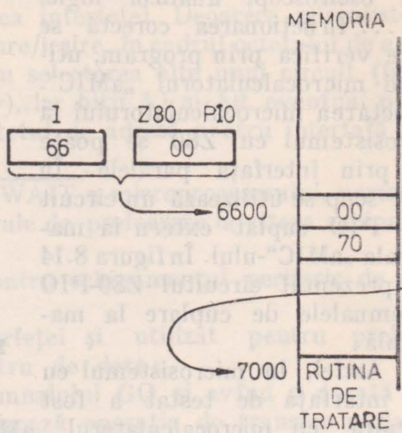


Fig. 8.18. Răspunsul la întreruperea generată de Z8Ø-PIO.

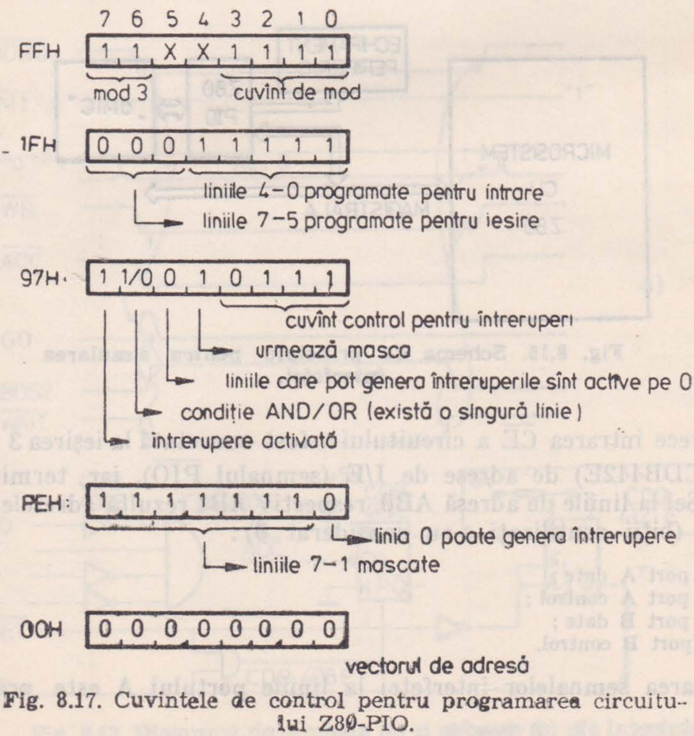


Fig. 8.17. Cuvintele de control pentru programarea circuitului Z8Ø-PIO.

— bucla de simulare;

— subrutina de tratare a întreruperii generate de activarea semnalului RESET (tehnica de răspuns prezentată schematic în figura 8.18).

```
; Programarea circuitului Z8Ø-PIO
        ORG   6ØØØH       ;
        LD    A, ØFFH     ; setează modul 3 (control bit)
        OUT   (61H), A    ; transmite în portul A control
        LD    A, 1FH      ; liniile 7, 6, 5 programate pentru
        OUT   (61H), A    ; ieşire şi liniile 4, 3, 2, 1, Ø pentru intrare
        LD    A, 97H      ; cuvîntul de control pentru întreruperi
        OUT   (61H), A
        LD    A, ØFEH     ; numai bitul Ø (RESET) poate genera
        OUT   (61H), A    ; întrerupere
        LD    A, Ø        ; vectorul de întreruperi
        OUT   (61H), A
; Iniţializarea sistemului de întreruperi la UCP
        IM2               ; setează modul 2 de răspuns la întreruperi
        LD    A, 66H      ; încarcă registrul I
        LD    I, A
        LD    HL, 7ØØØH   ; adresa subrutinei de tratare a întreruperii
        LD    (66ØØH), HL ; memorează la adresa 66ØØH şi 66Ø1H
; Iniţializare semnale de ieşire PIO
SIMUL : LD    A, 8ØH      ; iniţializare port A
        OUT   (6ØH), A    ; WAIT=1, GO=Ø, BUSY=Ø
        EI                ; activarea sistemului de întreruperi
; Bucla de simulare a interfeţei
; Aşteaptă selectarea interfeţei
BUCLA : IN    A, (6ØH)    ; citeşte liniile portului A
        AND   1EH         ; selectează biţii 4, 3, 2, 1
        XOR   4           ; testează semnalele A5, WR, M1, IORQ
        JR    NZ, BUCLA   ; dacă cel puţin un bit nu este activ aşteaptă
; Interfaţa a fost selectată, ACC=Ø
        IN    A, (6ØH)    ; citeşte portul A
        SET   5, A        ; poziţionează GO=1
        OUT   (6ØH), A    ; înscrie în portul A
; Trecerea în 1 a semnalului GO declanşează monostabilul 74121
        SET   6, A        ; poziţionează BUSY=1
        OUT   (6ØH), A    ; înscrie în portul A
; Activarea lui BUSY, resetează semnalul GO
        RES   5, A        ; poziţionează GO=Ø
        OUT   (6ØH), A    ; înscrie în portul A
; Întîrziere de Ø,5 ms, timp necesar pentru terminarea operaţiei de ieşire
        LD    A, 3DH      ; constanta de întîrziere pentru
DELAY : DEC   A           ; frecvenţa ceasului 2MHz
        JR    NZ, DELAY
; La sfîrşit dezactivează BUSY şi WAIT
        IN    A, (6ØH)    ; citeşte portul A
        SET   7, A        ; WAIT=1
        RES   6, A        ; BUSY=Ø
        OUT   (6ØH), A    ; înscrie în portul A
; Aşteaptă terminarea accesului
STOP :  IN    A, (6ØH)    ; citeşte portul A
        AND   1EH         ; selectează biţii 4, 3, 2, 1
        XOR   4           ; inversează M1 (semnale active pe Ø)
```

```
        JR      Z, STOP       ; dacă ACC=0, aşteaptă
        JR      LOOP          ; dacă ACC=1, reia secvenţa
; Subrutina de tratare a întreruperii generate
; de activarea semnalului RESET
        ORG     7000H
        EX      AF, AF'       ; salvare stare program
        EXX                   ; numai dacă este necesar — în acest exemplu ; nu este necesar
        POP     HL            ; înlocuieşte adresa de revenire din
        LD      HL, SIMUL     ; stivă cu adresa de relansare a
        PUSH    HL            ; programului de simulare
        EX      AF, AF'       ; aceste două instrucţiuni nu sînt
        EXX                   ; necesare — în general sînt utilizate pentru refacerea stării
                              ; programului întrerupt
        RETI                  ; return
        END
```

8.7. Cuplarea la microcalculator a unei miniimprimante MIM40

Miniimprimanta MIM40 (produsă la Electromureş Tg. Mureş) este o imprimantă paralelă cu 40 de caractere într-un rînd, fiecare coloană de caracter fiind formată din 7 puncte, fiecărui punct îi corespunde în sistemul electric de imprimare un ac de imprimare.

Formatul unui caracter este dat în figura 8.19 (7 linii × 5 coloane). Fiecare punct din matricea de imprimare este controlat prin software. Din punct de vedere mecanic, transportul capului de imprimare se realizează cu un tambur dispunînd de un ghidaj elicoidal ca în fig. 8.20.

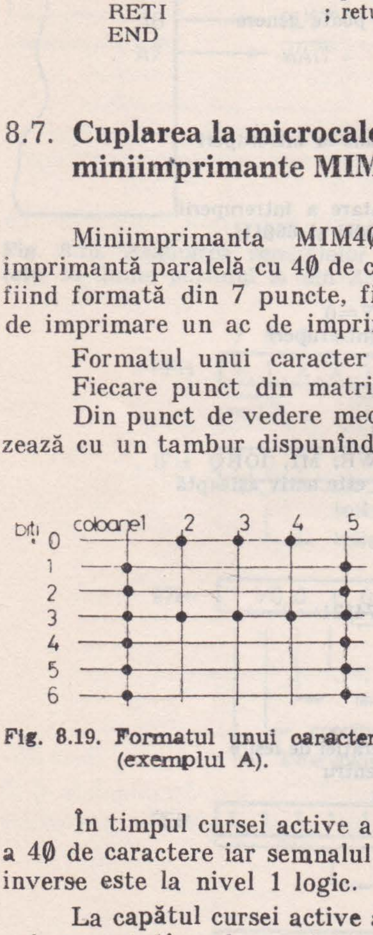


Fig. 8.19. Formatul unui caracter (exemplul A).

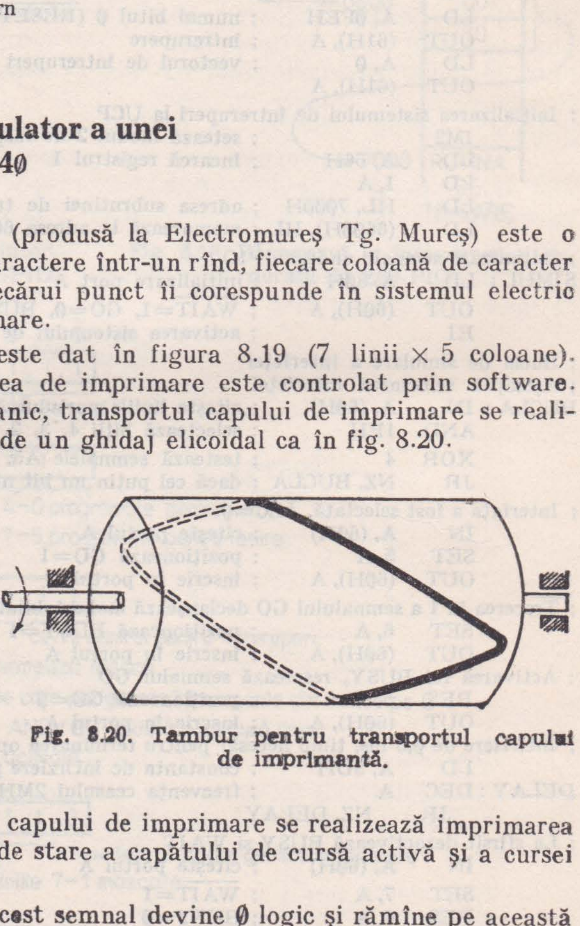


Fig. 8.20. Tambur pentru transportul capului de imprimantă.

În timpul cursei active a capului de imprimare se realizează imprimarea a 40 de caractere iar semnalul de stare a capătului de cursă activă şi a cursei inverse este la nivel 1 logic.

La capătul cursei active acest semnal devine 0 logic şi rămîne pe această valoare tot timpul cursei inverse. Semnalul de stare este generat printr-un sistem mecanic rigidizat cu tamburul.

Schema interfeţei de cuplare a microcalculatorului aMIC cu miniimprimanta este prezentată în figura 8.21.

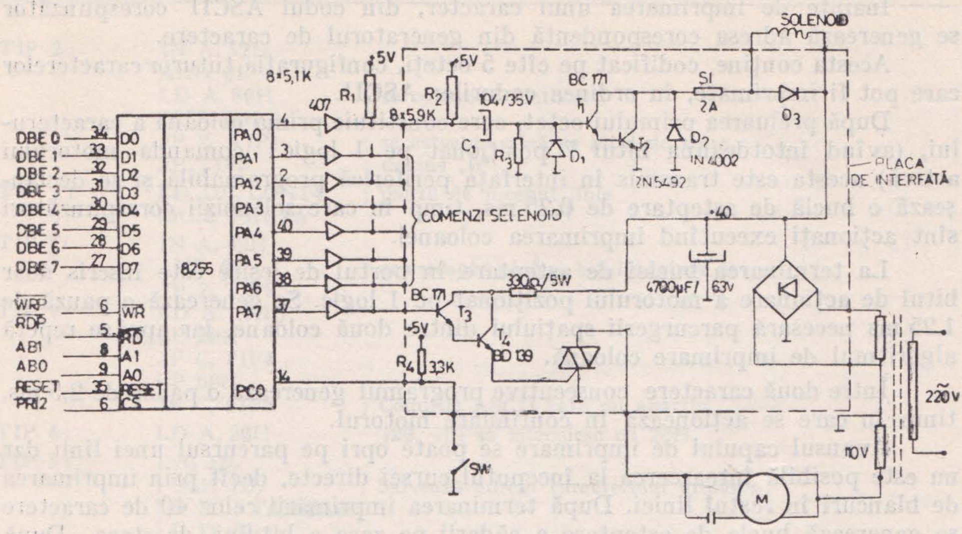


Fig. 8.21. Schema interfeței de cuplare a microcalculatorului cu miniimprimanta MIM4Ø.

Descrierea interfeței

Miniimprimanta este cuplată la microcalculator prin intermediul unei interfețe periferice programabile 8255, care comandă prin software schema electrică de imprimare și preia starea curentă de poziție a tamburului.

Portul A al interfeței periferice programat ca ieșire asigură cele 7 semnale de comandă a solenoizilor acelor de imprimare precum și semnalul de comandă a motorului care acționează tamburul.

Comanda solenoidului se realizează printr-un etaj de comutare, realizat cu un montaj Darlington, prevăzut cu protecție la tensiunile inverse induse în solenoid. (dioda D2).

Comanda acționării motorului este realizată printr-un montaj Darlington, care generează semnalul de comandă, pentru un triac care asigură excitația motorului la 11Ø V c.a.

Imprimarea unei linii decurge astfel :

Întrucît poziția capului de imprimare poate fi presupusă oarecare, comanda de acționare a motorului trebuie să asigure poziționarea la capăt de rînd a capului înaintea imprimării. Astfel, dacă semnalul de stare este pe 1 logic, se comandă rotirea motorului pînă la anularea acestui semnal, parcurgîndu-se astfel restul de cursă directă pînă la capătul de cursă. Cu motorul acționat programul de comandă asigură o buclă de așteptare de aproximativ 7 ms timp în care capul de imprimare ajunge la începutul cursei directe.

Se testează din nou semnalul de stare și dacă nu a devenit încă 1 logic se așteaptă modificarea sa (cu motorul acționat). Programul de imprimare presupune existența într-o zonă tampon de date a celor 4Ø coduri ASCII corespunzătoare caracterelor care trebuie imprimate.

Înainte de imprimarea unui caracter, din codul ASCII corespunzător se generează adresa corespondentă din generatorul de caractere.

Acesta conține, codificat pe cîte 5 octeți, configurația tuturor caracterelor care pot fi imprimate, în ordinea codurilor ASCII.

După preluarea primului octet, care constituie prima coloană a caracterului, (avînd întotdeauna bitul 7 poziționat pe 1 logic : comanda motorului activă) acesta este transmis în interfața periferică programabilă și se declanșează o buclă de așteptare de 0,25 ms, timp în care solenoizii corespunzători sînt acționați executînd imprimarea coloanei.

La terminarea buclei de așteptare în portul de ieșire este înscris doar bitul de acționare a motorului poziționat pe 1 logic. Se generează o pauză de 1,25 ms necesară parcurgerii spațiului dintre două coloane, iar apoi se repetă algoritmul de imprimare coloană.

Între două caractere consecutive programul generează o pauză de 2,5 ms, timp în care se acționează în continuare motorul.

Avansul capului de imprimare se poate opri pe parcursul unei linii dar nu este posibilă întoarcerea la începutul cursei directe, decît prin imprimarea de blancuri în restul liniei. După terminarea imprimării celor 40 de caractere se generează bucla de așteptare a căderii pe zero a bitului de stare. După aceasta se generează o pauză de ordinul 130 ms acoperitoare pentru cursa inversă, iar apoi se oprește comanda motorului.

Durata pauzelor introduse în program sînt astfel alese încît imprimarea a 4 puncte alăturate să genereze un pătrat.

Micșorînd pauza între imprimarea a două coloane succesive se poate mări numărul de caractere imprimate pe o linie.

Spațiul între două linii succesive este generat automat prin forma ghidajului elicoidal de pe tambur, deci distanța între două linii imprimate este întotdeauna aceeași.

Prezentarea programului de imprimare

Înainte de lansarea subrutinei de imprimare a unei linii, programul de imprimare trebuie să asigure încărcarea unei zone tampon de date avînd lungimea de 40 de octeți, cîte un octet pentru fiecare cod ASCII care urmează să fie imprimat. După încărcarea zonei tampon, subrutina de imprimare poate fi executată iar pentru imprimarea unei pagini se repetă secvența de operații : încărcarea zonă tampon — execuție subrutină imprimare linie.

Subrutina de imprimare ia în considerare numai coduri cuprinse între 20H și 5FH pentru alte coduri imprimînd blanc. Generatorul de caractere conține cifrele zecimale, literele majuscule de la A la Z și următoarele caractere speciale : blanc, ?, „ “, , $, ,', (,), *, +, ', —, ., /, ;, , =, ,?.

Subrutina de afișare a unei linii este prezentată în cele ce urmează :

```
IMPRIM:  PUSH H
         LD A, 81H
         OUT 43H         ; programarea interfeţei 8255
         XOR A
         OUT 42H
         LD HL, BUFTIP   ; HL-adresa zonei tampon
         LD B, 28H       ; 40 caractere pe linie
```

```
TIP 2 :   IN A, 42H
          AND Ø1
          LD A, 8ØH          ; se comandă motorul
          OUT 4ØH A
          JP Z, TIP1         ; se aşteaptă pînă ce bitul de stare
          JP TIP2            ; devine „Ø"
TIP 1 :   LD DE, CONT4       ; pauză de început linie
          CALL PAUZA
TIP 3 :   IN A, 42H
          AND Ø1             ; se aşteaptă pînă ce bitul de stare devine „1"
          JP Z, TIP3
TIP 7 :   LD A, (HL)
          CP 2ØH
          JP C, TIP4
          CP 6ØH
          JP C, TIP5         ; Codurile ASCII care nu aparţin
TIP 4 :   LD A, 2ØH            [2Ø, 5F] se înlocuiesc cu 2ØH.
TIP 5 :   INC HL
          PUSH HL            ; Salvează adresa caracterului următor
          LD HL, TAB-ØAØF
          LD D, ØØ
          LD E, A
          ADD HL, DE
          ADD HL, DE
          ADD HL, DE
          ADD HL, DE         ; Calculul adresei din generatorul de caractere a codului
          ADD HL, DE         ; din Acc.
          LD C, Ø5H          ; Nr. de coloane într-un caracter
TIP 6 :   LD A, (HL)         ; Se încarcă 8255 cu o coloană de imprimare
          OUT 4ØH A
          LD DE, CONT1
          CALL PAUZA         ; durata de acţionare a solenoizilor
          LD A, 8ØH
          OUT 4ØH A          ; comandă motor
          LD DE, CONT2
          CALL PAUZA         ; pauză între 2 coloane succesive
          INC HL
          DEC C
          JP NZ, TIP6
          LD DE, CONT3       ; pauză între 2 caractere
          CALL PAUZA
          POP HL
          DEC B              ; dacă mai sînt caractere de imprimat
          JP NZ, TIP7
TIP 8 :   IN A, 42H
          AND Ø1             ; se aşteaptă ca bitul de stare să devină „Ø"
          JP NZ, TIP8
          LD DE, CONI5       ; pauză necesară cursei inverse
          CALL PAUZA
          XOR A              ; oprire motor.
          OUT 4ØH A
          POP H
          RET
PAUZA :   DEC DE
          LD A, E
     D    OR
          JP NZ, PAZUA
          RET.
```

Configuraţia semnalelor la conectorul de interfaţă cu miniimprimanta

1. PA7 bufferat	9. PB7	17. PC7
2. PA6 bufferat	10. PN6	18. PC6
3. PA5 bufferat	11. PB5	19. PC5
4. PA4 bufferat	12. PB4	20. PC4
5. PA3 bufferat	13. PB3	21. PC3
6. PA2 bufferat	14. PB2	22. PC2
7. PA1 bufferat	15. PB1	23. PC1
8. PA∅ bufferat	16. PB∅	24. PC∅
		25. GND.

8.8. Cuplarea microcalculatorului cu un programator de EPROM

Programatorul asigură posibilităţi de programare pentru circuitele EPROM de tipul Intel 2716.

Configuraţia circuitului este următoarea (fig. 8.22).

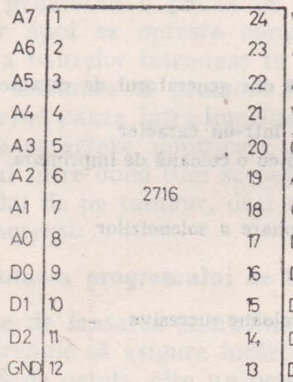


Fig. 8.22. Configuraţia terminalelor circuitului 2716.

Semnificaţia semnalelor :

D∅—D7 = date intrare/ieşire

A∅—A1∅ = adrese

Vpp = selecţie regim de programare

$\overline{OE}$ = validare ieşire

$\overline{CE}$ = intrare selecţie circuit

Modurile de lucru ale circuitului sînt prezentate în figura 8.23 a, unde se pun în evidenţă condiţiile în care se realizează fiecare regim : citire, programare, verificare.

Regimul de programare, verificare este selectat forţînd la intrarea Vpp o tensiune de 25 V.

Datele de intrare trebuie să fie stabile înainte şi după pulsul de programare, minimum 2 μs. Pulsul de programare durează între 45 şi 55 μs şi este livrat circuitului la intrarea $\overline{CE}$ de nivel TTL (fig. 8.23 b).

PIN / MOD	$\overline{CE}$/PGM (18)	$\overline{OE}$ (20)	Vpp (21)	Vcc (24)	OUTPUTS (9–11,13–17)
READ	V_{IL}	V_{IL}	+5	+5	D_{OUT}
STANDBY	V_{IH}	NU CONT	+5	+5	HIGH Z
PROGRAM.	PULS V_{IL} V_{IH}	V_{IH}	+25	+5	D_{IN}
PROGR. VERIFY	V_{IL}	V_{IL}	+25	+5	D_{OUT}
PROGR. INHIB.	V_{IL}	V_{IH}	+25	+5	HIGH Z

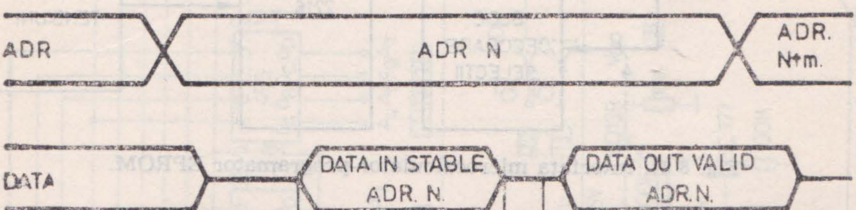


Fig. 8.23. Memoria EPROM 2716. Regimurile de lucru (a), diagramele de timp (b).

În figura 8.24 este prezentată schema bloc de interfaţare a programatorului la microcalculator.

Interfaţarea este realizată utilizîndu-se adresarea diferitelor operaţii ca porturi. Atît datele, adresele cît şi comenzile sînt livrate programatorului prin interfaţa periferică programabilă, fiind memorate în registre cu încărcare paralelă (CDB495).

Descrierea funcţionării programatorului (fig. 8.25)

Dispozitivul este cuplat la magistrala externă de date a microcalculatorului, necesitînd semnale de adresă (AB3, AB4) şi semnale de comandă (RDB, IOREQ).

Semnalele de adrese şi comenzi sînt folosite pentru decodarea celor 6 selecţii necesare funcţionării programatorului. Transferul de date din/înspre

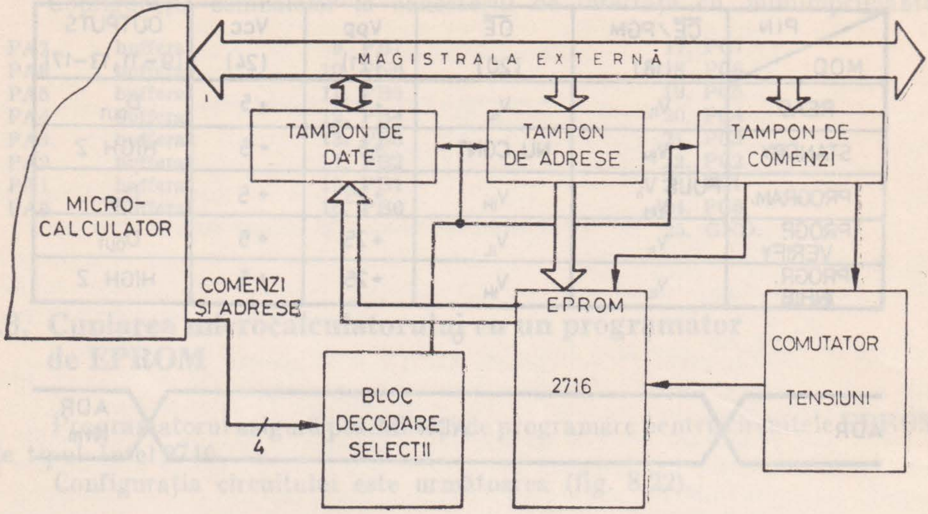


Fig. 8.24. Interfaţa microcalculator-programator EPROM.

programator se realizează prin instrucţiuni de intrare/ieşire. Codurile porturilor I/O sînt astfel construite încît selectează magistrala externă de date prin cuartetul superior şi porturile din programator prin cuartetul inferior.

Adresele sînt prezentate programatorului octet cu octet fiind încărcate în registrele tampon de adrese iar datele sînt înscrise în registrul tampon de date sau citite din EPROM prin bufferele de separare de tipul 74Ø9.

Pentru protejarea memoriei EPROM ieşirile registrului tampon de date au fost bufferate cu circuite „trei stări" de tipul 74125.

Pe pinul de alimentare Vcc al circuitului EPROM se aplică tensiune de +5 V programată. În timpul introducerii respectiv scoaterii circuitului 2716 din soclu tensiunea de alimentare nu ajunge la circuit.

Deschiderea liniilor de ieşire din tamponul de date spre circuitul EPROM se comandă prin înscrierea unui bit corespunzător într-un registru de comenzi.

Întrucît, în funcţie de regimul de lucru, intrarea Vpp a circuitului 2716 este forţată pe 5 V (în regim de citire) respectiv 25 V (în regim de programare/verificare) s-a prevăzut un circuit de comutare nivele de tensiune comandat tot printr-un bit înscris în registrul de comenzi.

În timpul operării cu memoria EPROM s-a prevăzut posibilitatea de preluare a stării tensiunilor Vpp, respectiv Vcc, prin citirea pe magistrala de date a unui cuvînt de stare. În funcţie de nivelul tensiunii Upp $\lesseqgtr$24 V bitul Ø din cuvîntul de stare este poziţionat pe 1 sau zero logic.

Bitul 1, din cuvîntul de stare este poziţionat pe 1 logic dacă EPROM-ul nu este alimentat cu tensiunea +5 V.

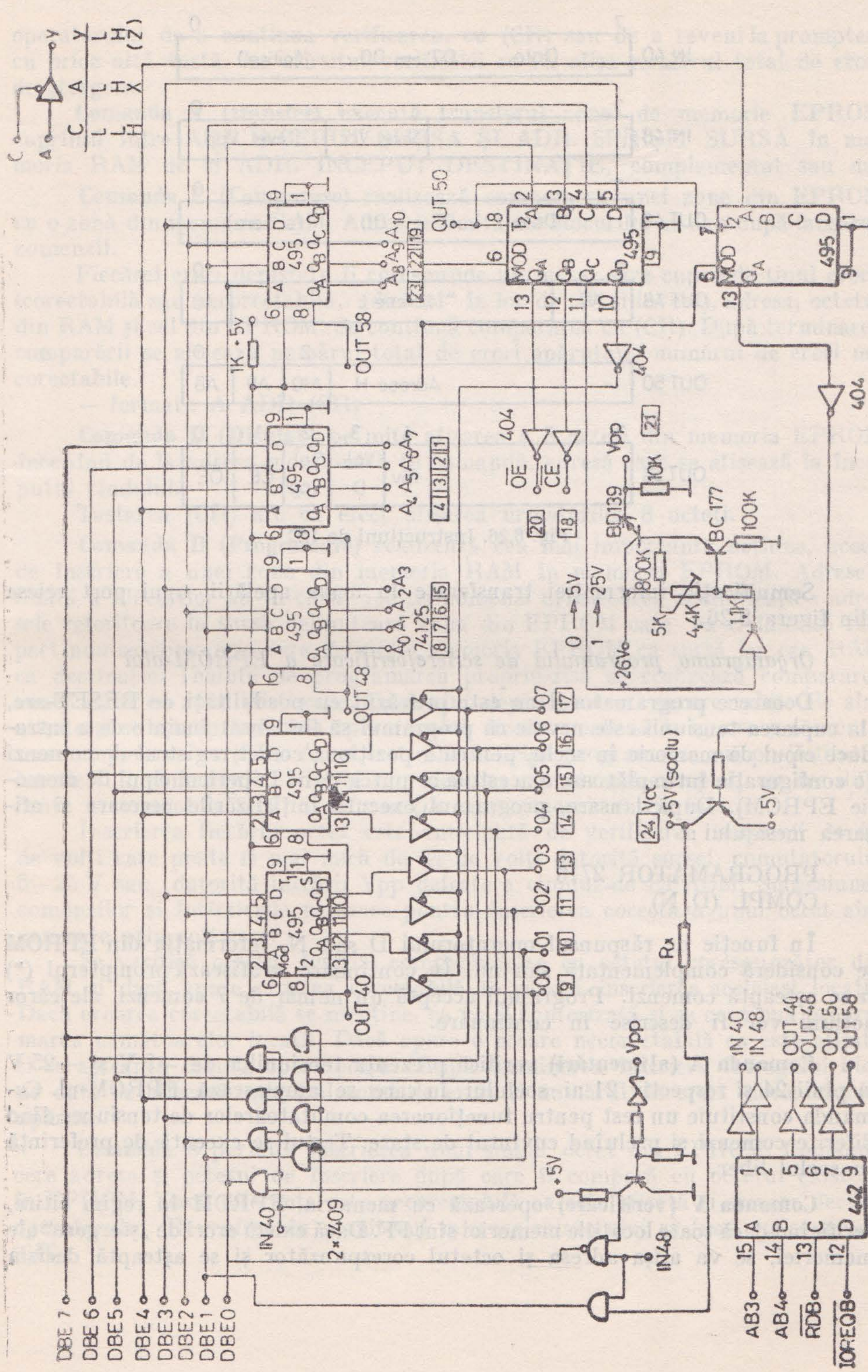


Fig. 3.25. Programator de memorii EPROM 2716.

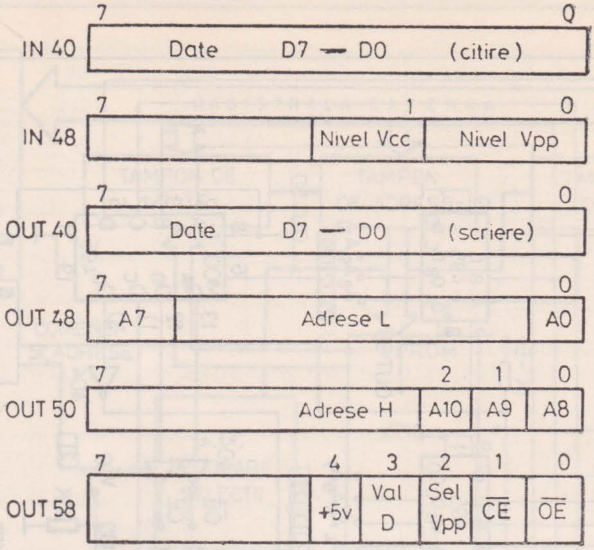


Fig. 8.26. Instrucţiuni de I/E.

Semnificaţia informaţiei transferate în urma apelării unui port reiese din figura 8.26.

Organigrama programului de scriere/verificare a EPROM-ului

Deoarece programatorul nu este prevăzut cu posibilităţi de RESET-are, la cuplarea tensiunii este nevoie ca programul să fie lansat înainte de a introduce cipul de memorie în soclu, pentru a poziţiona corect registrul de comenzi (o configuraţie întîmplătoare a acestuia ar putea pune în pericol cipul de memorie EPROM). După lansare, programul execută iniţializările necesare şi afişarea mesajului :

PROGRAMATOR_2716
COMPL (D, N)

În funcţie de răspunsul operatorului D sau N, informaţia din EPROM se consideră complementată sau nu. În continuare se afişează prompterul (*) şi se aşteaptă comenzi. Programul acceptă un număr de 7 comenzi, ale căror acţiuni vor fi descrise în continuare.

Comanda A (alimentări) verifică prezenţa tensiunilor de +5 V şi +25 V la pinii 24 şi respectiv 21 ai soclului în care se amplasează EPROM-ul. Comanda constituie un test pentru funcţionarea comutatoarelor de tensiune, dînd diferite comenzi şi preluînd cuvîntul de stare. Testul se execută de preferinţă cu soclul liber.

Comanda V (verificare) operează cu memoria EPROM în regim citire, verificînd dacă toate locaţiile memoriei sînt FF. Dacă există erori de „ştergere" ale memoriei, se va afişa adresa şi octetul corespunzător şi se aşteaptă decizia

operatorului de a continua verificarea, cu (CR) sau de a reveni la prompter, cu orice altă tastă. La sfîrşitul verificării se va afişa numărul total de erori de ştergere.

Comanda T (transfer) execută transferul zonei de memorie EPROM cuprinsă între ADR.ÎNCEPUT SURSA ŞI ADR. SFÎRŞIT SURSA în memoria RAM de la ADR. ÎNCEPUT DESTINAŢIE, complementat sau nu.

Comanda C (Comparare) realizează compararea unei zone din EPROM cu o zonă din memoria RAM. Adresele limită ale blocurilor se dau după lansarea comenzii.

Fiecărei erori depistate îi corespunde un mesaj care cuprinde tipul erorii (corectabilă sau necorectabilă, adică „1" în loc de „Ø" şi invers), adresa, octetul din RAM şi cel din EPROM. Se continuă compararea cu (CR). După terminarea comparării se afişează numărul total de erori apărute şi numărul de erori necorectabile.

— format : A ADR (CR)

Comanda D (Display) permite afişarea a 8 octeţi din memoria EPROM începînd de la adresa menţionată în comandă, adresă care se afişează la începutul rîndului.

Tastarea (CR) are ca efect afişarea următorilor 8 octeţi.

Comanda D (Programare) realizează cea mai importantă acţiune, aceea de înscriere a unei zone din memoria RAM în memoria EPROM. Adresele limită a blocurilor au în cazul acestei comenzi următoarea semnificaţie : adresele referitoare la sursă delimitează zona din EPROM care va fi înscrisă respectîndu-se convenţia care defineşte memoria EPROM ca sursă iar cea RAM ca destinaţie. Înainte de programarea propriu-zisă se realizează compararea zonei sursă cu zona destinaţie. Dacă sînt depistate erori necorectabile ele sînt afişate şi se aşteaptă decizia operatorului. Acesta poate să le ignore şi să continue compararea cu (CR). Acest test care anticipă programarea, dă posibilitatea înscrierii unei memorii EPROM care deşi nu este ştearsă complet, nu are diferenţe necorectabile faţă de informaţia care urmează să fie înscrisă.

Înscrierea fiecărui octet este anticipată de verificarea tensiunii de 25 de volţi care poate fi mai mică de 24 de volţi datorită sursei, comutatorului 5—25 V sau datorită intrării Vpp defecte a cipului de EPROM. Succesiunea comenzilor şi întîrzierile necesare pentru înscrierea corectă a unui octet sînt generate prin software.

Se verifică octetul înscris, comparîndu-se cu octetul corespunzător din RAM şi dacă apare eroarea corectabilă se repetă înscrierea aceleiaşi locaţii. Dacă eroarea corectabilă se menţine, ea va fi contorizată şi se continuă programarea următoarelor locaţii. Dacă apare o eroare necorectabilă ea este afişată şi se aşteaptă decizia operatorului. După terminarea înscrierii întregului bloc de date se afişează numărul total de erori şi numărul de erori necorectabile apărute.

Comanda S permite înscrierea unui singur octet din EPROM. Comanda cere adresa şi octetul de înscriere după care îl compară cu octetul existent în EPROM. Dacă diferenţa este necorectabilă ea este afişată şi se cere decizia operatorului. Revenirea în monitorul microcalculatorului se face cu comanda (CR).

8.9. Cuplarea cu un terminal DAF 2010

Prin intermediul interfeţei seriale 8251 se realizează transferul bidirecţional asincron de date între microcalculator şi un sistem DAF 2Ø1Ø, cu o rată de transfer de 24ØØ Bauds.

Informaţia vehiculată poate fi transferată octet cu octet, sau sub formă, 2de blocuri de date, înformat HEX-INTELLEC, care presupune următoarea structură a înregistrării :

CR LF : NO ADR T Date Control

unde : NO este numărul de octeţi de date ai înregistrării,
ADR — adresa de început a înregistrării.
T — tipul înregistrării.
Date — reprezintă maxim 16 octeţi
Control este suma logică a octeţilor din înregistrare.

Sfîrşitul de fişier este constituit de o înregistrare cu număr de octeţi zero iar adresa corespunzătoare va fi adresa de salt în fişierul încărcat.

Programul de transmisie a unui fişier, la lansare aşteaptă adresa de început şi sfîrşit a blocului de date iar la sfîrşitul transmisiei formează automat înregistrarea de sfîrşit fişier.

Programul de recepţie a unui fişier, aşteaptă după lansare, adresa de deplasare a blocului de date faţă de adresa de început a fişierului (intrinsecă).

Secvenţa de programare a interfeţei serie 8251 este prezentată în continuare :

```
MVI A, ØEAH        MVI A, 15H
OUT Ø1H            OUT Ø1H
```

Semnificaţia este următoarea : 2 biţi de stop, control de paritate dezactivat caracter pe 7 biţi, viteza de transmisie = rata de transmisie X16, activare transmisie şi recepţie, anularea erorilor din registrul de stare.

Subrutina de transmisie a unui caracter aflat în memorie la adresa HL este prezentată în continuare :

```
TCAR : MOV A, M        TC1 : IN Ø1H
       OUT ØØH               ANI Ø1H
                             JZ TC1
                             RET
```

Recepţia unui octet de la interfaţa serială este realizată de următoarea subrutină :

```
RCAR : IN Ø1H          IN ØØH
       ANI Ø2H         ANI 7FH
       JZ RCAR         RET
```

Programul de transmisie fişier preia din memorie blocuri de 16 octeţi, ormatează informaţia în HEX-INTELLEC şi calculează suma de control ealizînd astfel o înregistrare. Acest ciclu este repetat pînă la terminarea blocului de date, completîndu-se fişierul cu înregistrarea nulă.

Programul de recepţie fişier transferă în memorie înregistrări succesive, prelucrînd secvenţial fiecare octet şi verificînd la sfîrşit suma de control.

În cazul apariţiei unei erori programul afişează mesajul : ER : adresă, unde adresa constituie începutul înregistrării eronate.

În momentul recepţiei unei înregistrări nule se face ieşirea în monitorul microcalculatorului.

În continuare se dă configuraţia semnalelor la conectorul de interfaţă cu un DAF 2Ø1Ø :

1. GND	4.	6. STRAP pt. DAF 9÷19 neutilizat
2. T×D	5. STRAP pt. DAF	7. GND
3. R×D		8. —
		20. —

8.10. Interfaţa cu un minirobot

Minirobotul de laborator M2,5 elaborat la „Electrotimiş" Timişoara este un robot experimental cu aplicabilitate în industria electronică pentru manipularea de piese şi componente de mici dimensiuni şi greutate redusă (2,5 Nm ; greutate piese 3ØØ g, deschiderea dispozitivului de prehensiune : 7 cm ;). Minirobotul dispune de 5 grade de libertate fiind prevăzut cu 5 motoare pas cu pas (2Ø W) acţionate în buclă deschisă.

Minirobotul este realizat constructiv cu elemente mecanice din material plastic, transmisia este asigurată prin fire, posedă un panou de învăţare şi o tastatură cu electronică de urmărire.

Mişcările pe care le poate executa acesta sînt :

— pivotarea (rotaţia în jurul axului principal)

— mişcările braţului şi antebraţului

— mişcările de flexie şi supinaţie a mîinii mecanice.

Minirobotul este interfaţat la microcalculator printr-o interfaţa periferică programabilă, avînd cele 3 porturi programate ca ieşiri.

Prin această interfaţă microcalculatorul asigură 8 comenzi de sens de mişcare şi 8 comenzi de tact repartizate fiecărui motor pas cu pas. (1 comandă de sens mişcare şi 1 comandă de tact la fiecare motor).

Comenzile microcalculatorului sînt prelucrate într-o interfaţă de acţionare pentru motoarele pas cu pas.

Viteza de execuţie a mişcărilor este controlată software, interfaţa de acţionare intervenind la depăşirea valorilor prescrise pentru semnalele de tact.

Controlul mişcării accelerate şi decelerate este asigurat tot din interfaţa de acţionare.

Programul de aplicaţie este scris într-un limbaj de nivel înalt, ARM BASIC, cu subrutine în limbaj maşină Z8Ø. Între posibilităţile software remarcăm posibilităţile de elaborare a unor modele matematice de transformări de coordonate, un formalism matricial, posibilităţi de comandă în coordonate carteziene etc. Comenzile specifice ARMBASIC au fost utilizate şi la simularea mişcărilor unui minirobot în cadrul a două programe de aplicaţie : un program

de învățare pas cu pas a execuției unei mișcări și un program de execuție a unei traiectorii impuse. Programul de învățare permite simularea acționării pas cu pas a motoarelor ; aducînd minirobotul în poziții succesive, la comenzi de operator.

Cel de-al doilea program realizează simularea mișcării mîinii minirobotului pe o traiectorie a cărei coordonate succesive sînt calculate secvențial de către microcalculator. Simularea minirobotului pe microcalculatorul aMIC este reflectată prin afișarea stilizată pe display a pozițiilor succesive ale minirobotului.

8.11. Echipament de testare pentru microsisteme orientate pe magistrală

Structura hardware a microcalculatorului „aMIC“ constituie procesorul „master“ al unui sistem biprocesor de test în curs de elaborare la ITC.

Sistemul de test asigură posibilități de fabricație, service și instalare de aplicații a microsistemelor cu circuite LSI organizate pe principiul magistralei.

Echipamentul realizează o testare funcțional dinamică pe principiul emulării.

Stimularea plachetelor sau microsistemelor sub test este asigurată de emulator iar evaluarea răspunsurilor se face prin sondă mobilă cu analiză de semnături.

Resursele sistemului LSIMINITEST sînt : memorie utilizator în procesorul master 48 Ko, memorie de emulare 8 Ko, sistem de operare și interpreter de test rezidente în EPROM, procesor master (16 Ko), monitor de depanare/testare (rezident sau încărcat în memoria externă) programe de test și module de test funcțional încărcate din memorie externă, program de condus sonda mobilă, monitor de emulare (2 Ko).

Memoria externă este realizată cu minifloppy sau casetă audio. Sistemul de test are ca display un receptor TV obișnuit, dispune de miniimprimantă MIM4Ø, are disponibilități de transmisie/recepție serială de informații, putînd constitui un post de lucru al unei echipament de test complex, multipost, LSI TEST.

Sistemul de testare are posibilități de punere la punct și de editare a programelor de test, avînd interfațat un programator de memorii EPROM.

Emulatorul dispune de următoarele facilități :

Stimularea în timp real a plachetelor sau microsistemelor sub test cu trei categorii de stimuli : primari, module de test funcționale și stimuli defavorabili, memorie de emulare relocabilă în spațiul de adresă al procesorului emulator, ceas de gardă, generare și tratare de întreruperi, memorie trasoare, emulare pe microsisteme cu microprocesor Z8Ø sau 8Ø8Ø.

POD-ul emulatorului este constituit din microprocesorul 8Ø8Ø sau Z8Ø, bufferele de interfață și controlerul acestora. Intervenția pe placa sau microsistemul sub test se face la nivelul DIP-ului pe soclul microprocesorului emulat. Un translator de bus asigură posibilitatea testării microsistemelor organizate pe diferite standarde de magistrală. El este compus dintr-un modul

de unitate centrală specific, un modul de memorie specific şi o placă adaptoare de bus. Sonda mobilă asigură facilităţi de ridicare a unei semnături, discriminează nivele TTL, MOS, analogice, lucrează sub comanda procesorului master şi emulatorului.

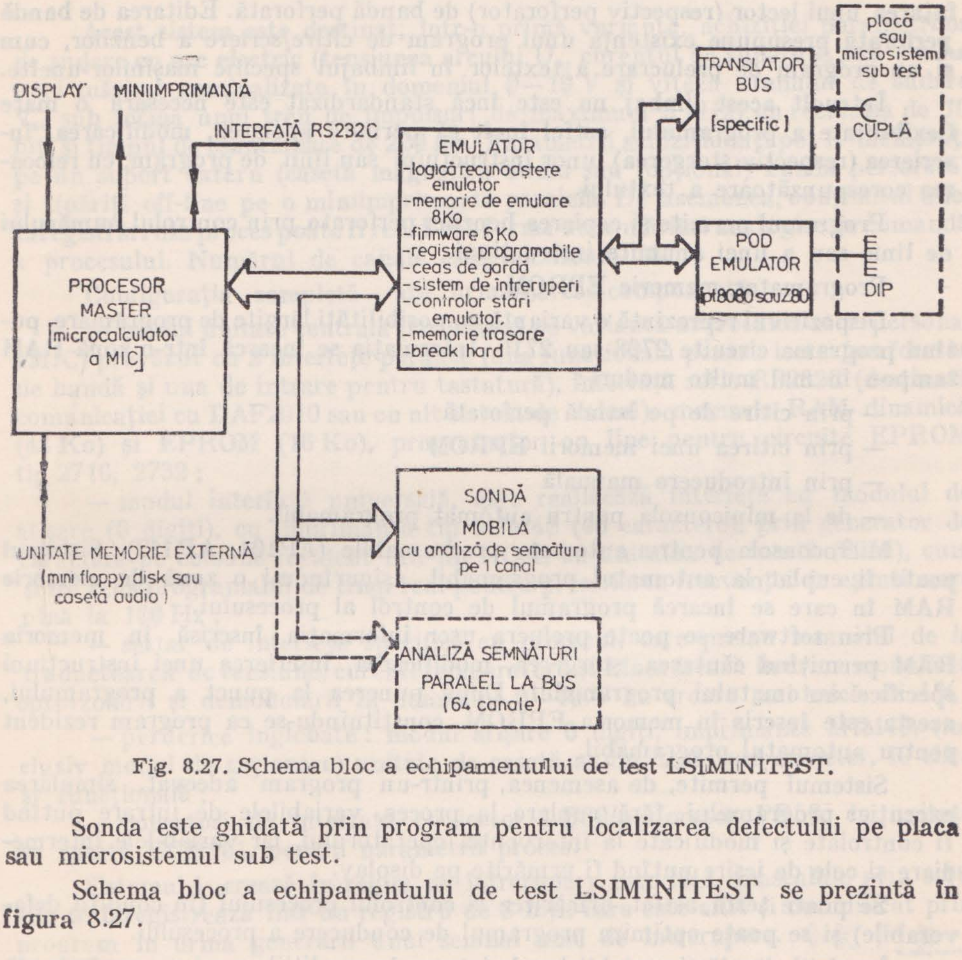


Fig. 8.27. Schema bloc a echipamentului de test LSIMINITEST.

Sonda este ghidată prin program pentru localizarea defectului pe placa sau microsistemul sub test.

Schema bloc a echipamentului de test LSIMINITEST se prezintă în figura 8.27.

8.12. aMIC-ul în unităţi de deservire pentru maşini-unelte

În cadrul unei lucrări în curs de elaborare la ITC în colaborare cu o întreprindere constructoare de maşini-unelte, microcalculatorul aMIC se constituie ca o unitate de deservire a maşinilor-unelte, preluînd următoarele funcţiuni :

— editare de bandă perforată ;

— programare EPROM ;

— miniconsolă pentru automate programabile ;
— imprimare programe.

Editor de bandă perforată

Resursele hardware ale microcalculatorului sînt completate cu interfaţarea unui lector (respectiv perforator) de bandă perforată. Editarea de bandă perforată presupune existenţa unui program de citire/scriere a benzilor, cum şi un program de prelucrare a textelor în limbajul specific maşinilor-unelte.

Întrucît acest limbaj nu este încă standardizat este necesară o mare flexibilitate a programului, astfel încît să permită apelarea, modificarea, înscrierea (respectiv ştergerea) unor instrucţiuni sau linii de program, cu relocarea corespunzătoare a textului.

Programul permite şi copierea benzilor perforate, prin controlul numărului de linie sau a unei anumite instrucţiuni.

Programator memorie EPROM

Dispozitivul reprezintă o variantă cu posibilităţi lărgite de programare, putînd programa circuite 27Ø8 sau 2716. Informaţia se încarcă într-o zonă RAM tampon în mai multe moduri :

— prin citire de pe bandă perforată
— prin citirea unei memorii EPROM
— prin introducere manuală
— de la miniconsola pentru automat programabil.

Microconsola pentru automate programabile (AP1Ø1, AP117). Sistemul poate fi cuplat la automatul programabil, asigurîndu-i o zonă de memorie RAM în care se încarcă programul de control al procesului.

Prin software se poate prelucra uşor informaţia înscrisă în memoria RAM permiţînd căutarea, ştergerea, modificarea, înscrierea unei instrucţiuni specifice automatului programabil. După punerea la punct a programului, acesta este înscris în memoria EPROM, constituindu-se ca program rezident pentru automatul programabil.

Sistemul permite, de asemenea, printr-un program adecvat, simularea execuţiei programului, fără cuplare la proces, variabilele de intrare putînd fi controlate şi modificate la intervenţia operatorului, iar variabilele intermediare şi cele de ieşire putînd fi urmărite pe display.

Se poate testa, astfel, urmărirea şi controlul procesului (în condiţii defavorabile) şi se poate optimiza programul de conducere a procesului.

În cazul simulării variabilelor de intrare în condiţiile unui proces în buclă închisă, se verifică optimalitatea programului de control al procesului înainte de realizarea aplicaţiei, sistemul preluînd funcţii de sistem expert.

Interfaţarea miniimprimantei MIM4Ø

În cadrul sistemului de deservire a maşinilor-unelte, funcţia de imprimare este utilă pentru a asigura un suport de ieşire al programelor în procesul prelucrării şi modificării acestora.

Soluţia de interfaţare a miniimprimantei la sistem a fost prezentată în paragraful 8.7.

8.13. Sistem de înregistrare/redare a parametrilor semicontinui de proces

Acest sistem este destinat, într-o primă variantă, achiziției parametrilor de sudare cu arc electric (tensiunea arcului U_a, curentul de sudare I_s sub forma unei mărimi nominalizate în domeniul Ø—1Ø V și viteza trenului de sudare V_s, sub forma unui tren de impulsuri de maximum 5 kHz) cu rezoluția de 10 biți și timpul de eșantionare de 2ØØ ms. Parametrii achiziționați pot fi memorați pe un suport extern (casetă magnetică audio sau (opțional) bandă perforată) și tipăriți off-line pe o miniimprimantă paralelă. De asemenea, conținutul unei înregistrări din proces poate fi redat sub forma unor mărimi analogice de comandă a procesului. Numărul de canale este maximum 16.

Configurația completă are următoarea componență:

— modul unitate centrală (structurat pe varianta microcalculator personal aMIC) prevăzut cu 2 interfețe paralele (una funcțională pentru lector/perforator de bandă și una de intrare pentru tastatură), interfață serie RS232C (destinată comunicației cu DAF2010 sau cu alt sistem de calcul), memorie RAM dinamică (48 Ko) și EPROM (16 Ko), programator on line pentru circuite EPROM tip 2716, 2732 ;

— modul interfață universală, care realizează interfețe cu modulul de afișare (6 digiți), cu imprimanta tip MIM4Ø (4Ø caractere), prin generator de caractere pe coloane rezident într-un ROM, cu unitatea de casetă (8251), cum și un ceas programabil de timp real pentru prescrierea frecvenței de eșantionare pînă la 1ØØ Hz ;

— sertar de interfețe specifice cu procesul, care preiau semnalul de la traductoarele de tensiune, curent, temperatură și-l încadrează în urma modulării, optoizolării și demodulării în domeniul Ø—1Ø V cu eroare mai mică de 1% ;

— periferice înglobate : modul afișare 6 digiți, imprimantă MIM4Ø, (inclusiv modul de acționare) unitate de casetă audio, tastatură caracter, 10 cifre și funcțională ;

— conectori de interfață cu lector LB5Ø/perforator PB5Ø, cu unitate de casetă UCM1Ø1, pentru parametrii proces.

Sistemul lucrează în regim de întreruperi, mod IM1, mascabile prin soft. Ele se înregistrează într-un registru de 8 biți, care este citit și decodificat prin program în urma generării unui semnal unic de întrerupere.

În regim de programare, operatorul poate prescrie prin dialog;

— numărul canalului ;

— tipul canalului ;

— tipul de eșantionare (1ØØ Hz) ;

— funcția înregistrată

În figura 8.28 se prezintă schema bloc a sistemului de înregistrare/redare a parametrilor de proces

Software-ul aplicației asigură urmărirea procesului în timp real, fiind realizat în limbaj mașină 8Ø8Ø (sau Z8Ø).

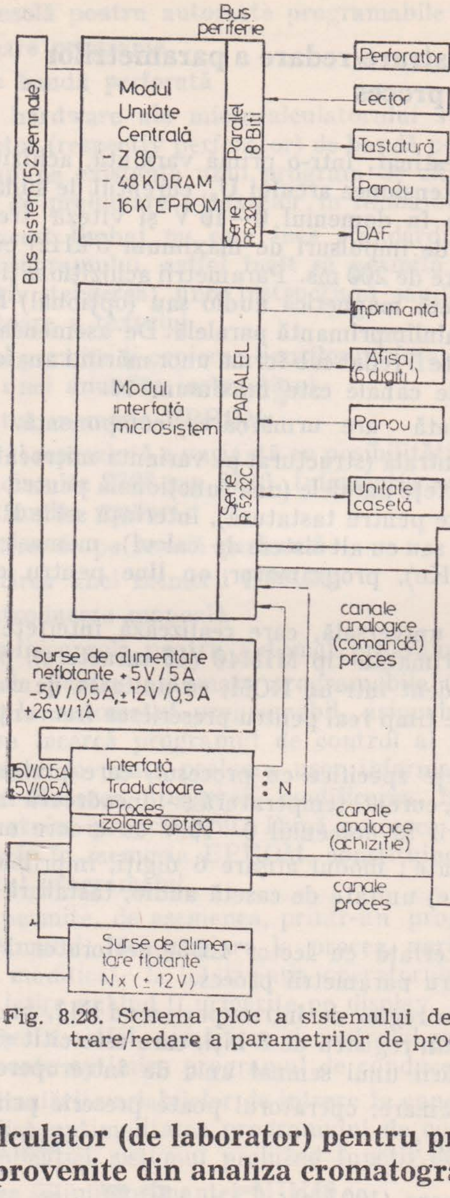


Fig. 8.28. Schema bloc a sistemului de înregistrare/redare a parametrilor de proces.

8.14. Microcalculator (de laborator) pentru prelucrarea datelor provenite din analiza cromatografică

Pentru a evita costul ridicat al unei implementări specifice şi pentru a putea destina sistemul şi altor proceduri de analiză fizico-chimică, soluţia aleasă la ITC Timişoara recurge la o variantă cu un anumit grad de modularitate, funcţia principală de sistem fiind preluată de microcalculatorul aMIC, cu resurse proprii de memorie 64 Ko (48 Ko RAM şi 16 Ko EPROM).

Acest modul interfațează tastatura de caractere și funcțională, un DAF sau monitor TV ca echipament extern de vizualizare și asigură o interfață de date paralelă.

Restul periferiei este tratat pe modulul de interfață care gestionează imprimanta paralelă de tip MIM4Ø, înglobată în echipament, subsistemul de achiziție a mărimii de analiză, un ceas real de 2Ø ms și ceasul de eșantionare.

Pe același modul sînt rezidente și registrul de întreruperi și măștile de întreruperi.

Conversația numeric-analogică de tip interactiv se realizează pe 15 biți, plus un bit de semn, în domeniul −1ØØ mV — +1 V, avînd perioada de eșantionare sincronizată cu frecvența rețelei, pentru rejecția zgomotelor de înaltă frecvență.

Semnalul este preluat diferențial pentru a reduce zgomotele induse prin masă.

Semnalul de analiză provenit de la cromatograf se prezintă sub forma unei succesiuni de vîrfuri (peak-uri).

Fiecare „peak" apare la un moment de timp care trebuie corect estimat, întrucît identifică tipul unui anumit component din amestecul de analiză.

Integrala „peak"-ului dă o informație cantitativă asupra componentului detectat, astfel încît, în final, se poate face o estimare exactă a compoziției procentuale a probei analizate.

Software-ul sistemului este scris în limbaj de asamblare 8Ø8Ø (16 Ko program și 12 Ko zonă de date, permițînd integrarea a 35Ø peak-uri).

Pachetul de programe este organizat modular și cuprinde un monitor cu facilități de autodepanare, task-uri de dialog, de integrare, de corecție, de calcul al raportului (precizia de calcul : 24 biți).

Capitolul 9.

Limbajul BASIC pentru microcalculatorul personal aMIC Manual practic

9.1. Introducere

BASIC (Beginners All-purpose Symbolic Instruction Code) reprezintă un limbaj conversaţional de nivel înalt. Pentru execuţia programelor de aplicaţie scrise în BASIC se folosesc cel puţin trei metode distincte : interpretarea, compilarea şi combinarea lor. Pachetul software, care foloseşte metoda combinată, poartă numele de compilator/interpretor sau pseudo-compilator.

Fiecare din metodele de execuţie a programelor de aplicaţii are avantajele şi dezavantajele ei. În general, alegerea metodei este determinată de aplicaţie, de utilizarea particulară a calculatorului.

În timpul execuţiei, un interpretor analizează fiecare instrucţiune, verifică existenţa erorilor, apoi efectuează funcţia BASIC solicitată. Pentru a executa funcţia respectivă, interpretorul traduce instrucţiunea dată în codul ASCII într-un număr de instrucţiuni maşină executabile, pe care le execută în continuare. Acest proces se repetă la fiecare instrucţiune BASIC, chiar dacă instrucţiunea respectivă apare de mai multe ori în cadrul unui ciclu.

Modul interpretativ de execuţie presupune existenţa în memoria calculatorului a întregului program BASIC-sursă, sub forma unei liste de linii marcate cu etichete numerice. Fiecare instrucţiune de ramificaţie impune ca interpretorul să caute într-o listă de numere de linii, pentru a găsi instrucţiunea dorită În aceeaşi manieră interpretorul va căuta o anumită variabilă în cadrul unei liste date.

Metoda bazată pe compilare presupune traducerea programului sursă, scris în BASIC, într-un fişier obiect relocabil. În continuare, un program de sistem, denumit Editor de legături, va înlănţui toate fişierele obiect relocabile produse, cît şi fişiere obiect extrase dintr-o bibliotecă, pentru a produce fişierul executabil în cod maşină. Programul obiect compilat, va fi executat de 25÷30 de ori mai repede decît se executa în maniera interpretativă acelaşi program BASIC sursă.

Compilarea şi editarea de legături trebuie repetată la fiecare modificare executată în programul sursă. În cazul interpretorului se va modifica cu ajutorul unor facilităţi de editare încorporate, linia respectivă din programul sursă, după care se va trece direct la execuţie.

Astfel, un programator poate utiliza un interpretor pentru a pune la punct un program sursă BASIC, după care poate utiliza un compilator compatibil, pentru a obţine un program obiect, care poate fi executat într-un timp foarte scurt, corespunzător aplicaţiei date.

În comparaţie cu un compilator, interpretorul este mult mai ieftin.

Metoda bazată pe pseudo-compilare asigură un compromis între viteză şi cost. Astfel, costul unui pseudo-compilator este apropiat de cel al unui interpretor, în timp ce viteza sa este de circa cinci ori mai mare decît a acestuia din urmă. Viteza relativă a pseudo-compilatorului depinde de instrucţiunile programului, de tipul de date şi de gradul în care codul rezultat necesită translatarea în timpul execuţiei.

Varianta de BASIC aleasă pentru aMIC are un caracter interpretativ, ceea ce o face extrem de atractivă pentru punerea la punct a programelor. Marea majoritate a aplicaţiilor pentru acest calculator nu intră în categoria aplicaţiilor de timp real.

În cazul în care se impune o viteză de execuţie mai mare, se poate folosi metoda apelării unor subrutine, scrise în limbaj de asamblare, pentru acele zone din program care trebuie executate cu viteză mare.

Asemenea situaţii apar în cadrul aplicaţiilor de colectare a datelor sau de conducere a unor procese. Pentru a realiza accesul la subrutine scrise în limbaj de asamblare se foloseşte instrucţiunea CALL, care include un nume de variabilă (un indicator la o celulă de memorie în care se află punctul de intrare în subrutina scrisă în limbaj de asamblare) şi dacă este necesar, o listă de parametri care trebuie transferaţi subrutinei.

Pentru punerea la punct a programelor în limbaj de asamblare se poate folosi programul de sistem MATE (Monitor-Asamblor-Text Editor), programul asamblat fiind stocat pe caseta magnetică. În continuare el va fi încărcat în memoria operativă a calculatorului o dată cu programul BASIC-sursă. Într-unul din paragrafele acestui capitol se dau exemple pentru ilustrarea acestei tehnici.

Calculatorul aMIC are disponibile două versiuni de interpretoare BASIC. Prima versiune reprezintă un subset al celei de-a doua, în sensul că nu posedă instrucţiuni matriceale, instrucţiuni de prelucrare grafică şi instrucţiunea CALL. Prima versiune necesită un spaţiu de 8 Ko în memoria PROM, în timp ce versiunea a doua necesită un spaţiu de 14 Ko. Se menţionează şi posibilitatea încărcării interpretorului BASIC de pe caseta magnetică, în scopul reducerii numărului de circuite integrate de memorie PROM al căror cost este relativ ridicat.

Operarea sistemului. Elemente introductive privind programarea în limbajul BASIC

În vederea unei sesiuni de lucru, calculatorul se conectează la intrarea de antenă a unui televizor şi la o tastatură alfanumerică *. Televizorul va fi folosit ca echipament de ieşire, în timp ce tastatura va juca rolul unui echipament de intrare.

* Amănuntele privind conectarea la un televizor comercial, cît şi tastatura sînt date în capitolul 3 al lucrării.

După alimentarea de la rețea a televizorului și selectarea canalului corespunzător se va alimenta și calculatorul, prin intermediul sursei proprii, și se va acționa tasta RESET. În acest moment pe ecranul televizorului va apărea un text, care indică faptul că sistemul se află sub controlul programului monitor. În continuare se pot introduce, de la tastatură, comenzi monitor (descrise în capitolul referitor la monitorul sistemului). Pentru a lansa BASIC-ul, trebuie acționate tastele B și RETURN. Pe ecran va apărea mesajul READY, indicînd posibilitatea introducerii unor comenzi și instrucțiuni BASIC. De exemplu, dacă se dorește calculul valorii unei expresii aritmetice : $3\emptyset-2\cdot 3^2+4:2\cdot 3$, se va introduce de la tastatură următoarea instrucțiune :

```
1Ø PRINT 3Ø - 2 * 3 ↑ 2+4/2 * 3
```

Se observă că instrucțiunea începe cu o etichetă numerică (10). La sfîrșitul introducerii instrucțiunii trebuie acționată tasta RETURN. În continuare linia (instrucțiunea) introdusă este depusă în memoria calculatorului, fără a se executa. Se constată că operațiile de înmulțire, împărțire și ridicare la putere sînt descrise prin simbolurile *,/și ↑ .

În memoria calculatorului pot fi introduse de la tastatură succesiv mai multe instrucțiuni, avînd numere de linie diferite. Un grup de instrucțiuni depuse în memorie poartă numele de program.

Pentru a executa o instrucțiune (un program) depusă în memoria calculatorului, trebuie dată comanda RUN (urmată de RETURN). Astfel, introducînd instrucțiunea PRINT, și apoi comanda RUN, calculatorul va afișa pe ecran rezultatul expresiei (18), prioritatea în execuția operatorilor fiind cea obișnuită (întîi ridicări la putere, apoi înmulțiri și împărțiri și în final adunări și scăderi). După tipărirea rezultatelor, pe ecran se va afișa un mesaj de eroare cu numărul 1, indicînd faptul că programul executat nu s-a terminat cu o instrucțiune de oprire a execuției (STOP sau END). Pentru a evita această eroare, se va introduce după PRINT o nouă instrucțiune : 2Ø END.

Executînd din nou programul (format din două instrucțiuni : 1Ø PRINT ... și 20 END) și folosind tot comanda RUN, nu se va mai obține mesajul de eroare.

În instrucțiunea PRINT pot apărea mai multe expresii (după necesități) separate prin virgulă. De exemplu instrucțiunea :

```
1Ø PRINT 1+7, 3, 5, 5*2 ↑ 3, 19/4
```

Va tipări cinci valori, cîte două pe un rînd separate cu spații. De notat că ecranul televizorului este împărțit în 32 de rînduri a 30 de caractere fiecare. În cazul folosirii virgulei în instrucțiunea PRINT valorile sînt afișate începînd din poziția 1 și apoi din poziția 15 a unui rînd, realizîndu-se o tabulare automată. Pentru a nu se lăsa spațiu (tabula) între valorile tipărite, se folosește separatorul ; între expresii.

Astfel, 1Ø PRINT 8 ; 15/2−1 ; 2 ↑ 7 ; 19 va tipări cele patru valori 8,6. 5, 128 și 19 pe același rînd (fără tabulare).

Pentru a memora o valoare (un număr) care va fi folosită ulterior în program, se va utiliza o variabilă literală, căreia i se va atribui valoarea respectivă.

Limbajul BASIC dispune în acest sens de instrucţiunea de atribuire LET, cu ajutorul căreia se stochează în memoria calculatorului mărimi ce pot fi utilizate ulterior. Astfel, folosind variabila X :

```
5 LET X=15
1Ø PRINT X
2Ø PRINT X/2, X↑2—X+1
3Ø PRINT X
4Ø STOP
```

valoarea tipărită de instrucţiunile 1Ø şi 3Ø este aceeaşi (15). Instrucţiunea 2Ø utilizează valoarea variabilei X, fără a o modifica.

9.2. Elementele limbajului BASIC

În continuare vor fi prezentate cîteva noţiuni necesare pentru descrierea unei probleme în limbajul BASIC. Cuvintele folosite în limbajul BASIC formează vocabularul limbajului. Ele se scriu după reguli precise, date de sintaxa limbajului.

Caracterele întrebuinţate în BASIC pentru alcătuirea cuvintelor sînt :
—literele mari ale alfabetului : A, B, C, . . ., Z
— cifrele : Ø, 1, . . ., 9
— caractere speciale : + — * / = “ > < ·, ↑ ( ) ; $

9.2.1. **Constante.** Constantele sînt de două tipuri : constante numerice şi constante şir sau texte. În continuare, în lipsa menţionării explicite, prin constantă ne vom referi la constante numerice. Constantele şir reprezintă orice şir de caractere introdus între ghilimele. De exemplu : “STUDENŢII HARNICI“, “INTRODUCEŢI DATELE“, “127·5“, etc.

Constantele numerice utilizate în BASIC sînt reale. Ele pot avea, de exemplu, următoarele exprimări :

25   123.45   +4   —Ø.36   +.7321

De asemenea, la fiecare număr de mai sus, se poate adăuga un exponent, utilizînd litera E. Exponentul este un număr întreg. El indică puterea lui 1Ø, cu care se înmulţeşte numărul.

Astfel, următoarele constante sînt corecte :

25E—1=2.5   123.45E12   —Ø.36E—15

Intern, constantele sînt reprezentate în formatul cu virgulă mobilă, pe patru octeţi. Primii trei octeţi conţin mantisa, subunitară şi normalizată, iar octetul patru conţine exponentul :

```
|              M               | S |        E        |
 Ø                           23  24               31
```

unde :

M este mantisa normalizată $0.5 \leqslant M < 1$
S este un bit care reprezintă semnul mantisei (1 pentru negativ).
E este exponentul în complement faţă de doi. E reprezintă puterea lui doi, cu care se înmulţeşte mantisa.

Conform cu această reprezentare internă, cel mai mic număr manipulat va fi (în modul) : 2.71051E-2Ø iar cel mai mare : 9.22337E18. Din reprezentare se observă că sînt păstrate aproximativ 7 cifre semnificative.

9.2.2. **Variabile.** Variabilele utilizate în BASIC pot fi, ca şi constantele, de două tipuri : variabile numerice (pe care le vom numi, pe scurt, variabile) şi variabile şir.

Variabilele şir reprezintă şiruri de caractere alfanumerice. Numele unei variabile şir este format dintr-o literă urmată de caracterul $.

Exemple : A $, B $

Se pot utiliza şi tablouri de variabile şir, tablourile fiind formate din mai multe variabile şir, cu aceeaşi lungime şi acelaşi nume.

Numărul şi lungimea variabilelor şir ce formează un tablou, trebuie declarate într-o instrucţiune DIM, înainte de utilizarea tabloului. Variabilele şir ce formează un tablou, vor fi specificate prin utilizarea indicilor. De exemplu, dacă A $ este un tablou de zece variabile şir, a cîte 5Ø de elemente (caractere) fiecare, atunci vom referi variabilele şir prin : A $ (1), A $ (2), ..., A $ (1Ø), iar tabloul A $ va fi declarat într-o instrucţiune :

DIM A $ (1Ø, 5Ø).

În unele aplicaţii, este necesară utilizarea unor porţiuni (subşiruri) dintr-un şir, desemnat de o variabilă şir. Pentru specificarea subşirurilor unei variabile şir, se foloseşte notaţia : (n_1 TO n_2) asociată numelui variabilei şir ; n_1 este indicele primului caracter al subşirului, iar n_2 este indicele ultimului caracter al subşirului din variabila şir. De exemplu, fie B $="123ABC", subşirul B $ (2TO5)="23AB".

Indicii n_1 şi n_2 pot fi constante, variabile sau expresii, valoarea lor fiind cuprinsă între 1 şi dimensiunea (lungimea) şirului. În cazul absenţei indicelui n_1, se consideră că subşirul începe cu primul caracter al şirului, de exemplu : B $ (TO4)="123A".

În cazul absenţei indicelui n_1 se consideră că subşirul se termină cu ultimul caracter al şirului, de exemplu : B $ (3TO)="3ABC". Pot lipsi ambii indici, caz în care subşirul este identic cu şirul (variabila şir) dat. Exemplu : B$ (TO)= =B $ ="123ABC".

Se pot utiliza subşiruri ale variabilelor şir ce formează un tablou de variabile şir.

De exemplu, pentru tabloul A $ (10, 50) format din zece şiruri de cîte 5Ø de caractere, se poate specifica subşirul A $ (3, 12 TO 43), format din elementele 12 la 43, ale variabilei şir A $ (3).

De remarcat că, în cazul unui tablou de variabile şir, trebuie selectată variabila şir din tabloul la care ne referim şi apoi specificat un subşir din această variabilă.

Pentru subşiruri formate dintr-un singur caracter, se poate folosi un singur indice, cel al caracterului selectat.

Astfel : B $ (4)=B $ (4TO4)="A" ; B $ (6)="C", pentru variabila B $ utilizată mai sus, iar în cazul tabloului A $, elementul (caracterul) al treilea din şirul şapte va fi specificat prin A $ (7, 3) sau A $ (7, 3TO3).

Variabilele numerice utilizate în BASIC sînt reale. Variabilele pot fi simple sau indexate. Variabilele simple sînt reprezentate fie printr-o literă, fie o literă şi o cifră.

Exemple : A, V1, Q7, A9

Variabilele indexate reprezintă elementele unui tablou (vector sau matrice). Identificatorul (numele) tabloului trebuie să fie compus dintr-o singură literă. Litera identificator de tablou poate coincide cu numele unei variabile simple (fără a produce conflict).

Tablourile, înainte de utilizare, trebuie declarate în instrucţiunea DIM (sau o instrucţiune MAT). Indicele poate fi o constantă, o variabilă sau o expresie şi trebuie să aibă valoarea :

1 ⩽ indice ⩽ dimensiunea declarată în DIM

Observaţie :

Dacă, în urma evaluării expresiei indice, nu se obţine o valoare întreagă, se va reţine partea întreagă a valorii obţinute.

Exemple: E (1,5) ; V (ABS (R)) ; A (I-3, J-K)

9.2.3. **Operatori.** *Operatori aritmetici :*

↑ ridicare la putere,
*, / înmulţire, împărţire,
—, + scădere, adunare.

Operatorii au fost scrişi în ordinea priorităţii în evaluare (↑ are prioritatea cea mai mare). Cînd se doreşte schimbarea priorităţii în evaluare, sau cînd există dubii, este bine să se utilizeze parantezele ; operaţiile din interiorul parantezelor vor fi executate înaintea celor din exterior.

Operatori relaţionali

Operatorii relaţionali sînt utilizaţi în instrucţiunea IF pentru a determina relaţia dintre valorile a două expresii :

operator	semnificaţie	operator	semnificaţie
=	egalitate	>= sau =>	mai mare sau egal,
>	mai mare	<= sau =<	mai mic sau egal,
<	mai mic	<> sau ><	neegalitate.

9.2.4. **Funcţii.** În alcătuirea expresiilor pot fi utilizate următoarele funcţii matematice :

SIN(x) — sinus de x, unde x este un unghi exprimat în radiani ;
COS(x) — cosinus de x, unde x este exprimat în radiani ;
TAN(x) — tangentă de x, unde x este exprimat în radiani ;
ATN(x) — arctangentă de x. Rezultatul este exprimat în radiani :

$$(-PI/2 < ATN(x) < PI/2) ;$$

LOG(x) — logaritm natural din x ;
EXP(x) — calculează e^x ;

Observaţie :

Constantele π şi e sînt definite intern, în interpretor, şi pot fi utilizate în expresii prin simbolurile PI şi respectiv EE.

SQR(x) – calculează rădăcina pătrată din x.

ABS(x) – calculează valoarea absolută a lui x.

INT(x) – calculează cel mai mare întreg $\leqslant$x.

RND(x) – calculează un număr aleator în intervalul (0, 1). Valoarea argumentului x nu are importanţă în calcul.

SGN(x) – returnează :

1, dacă x > ∅

∅, dacă x=∅

–1, dacă x < ∅.

În toate funcţiile descrise, argumentul x poate fi în general o expresie oarecare.

Argumentele funcţiilor LOG şi SQR trebuie să fie numere pozitive, în caz contrar sistemul va răspunde printr-un mesaj de eroare.

În situaţia cînd, în calculul unei funcţii (expresii), rezultatul depăşeşte scala numerelor reprezentabile în calculator (2.71∅51E-2∅, 9.22337E18), se afişează la consolă un mesaj de eroare (UNDERFLOW sau OVERFLOW IN LINE NN), execuţia programului continuînd. Ca rezultat al funcţiei (expresiei) se ia cea mai mică, respectiv cea mai mare valoare reprezentabilă (corespunzător depăşirii). De pildă, în cazul funcţiei EXP(x), valoarea argumentului trebuie să fie cuprinsă aproximativ între $-45 < x < 44$, pentru ca rezultatul e^x să poată fi reprezentat intern.

Funcţia INT calculează cel mai mare întreg, mai mic, sau egal cu argumentul.

Astfel : INT(7.25)=7

INT(–7.25)=–8

INT(–.1)=–1

INT poate fi utilizată pentru a rotunji un număr la cel mai apropiat întreg.

Astfel, INT(x+∅.5) va calcula întregul cel mai apropiat de x.

Pe lîngă funcţiile descrise mai sînt disponibile : funcţii pentru lucrul cu şiruri de caractere, precum şi funcţii speciale de intrare/ieşire.

Astfel, pentru lucrul cu şiruri de caractere, se pot utiliza funcţiile :

VAL(şir) – calculează valoarea numerică a şirului, tratat ca o expresie aritmetică. De exemplu : VAL("127.5-B")=7.5, dacă variabila B are valoarea 12∅.

LEN(şir) – calculează lungimea şirului specificat. De exemplu : LEN("ABC123")=6, sau dacă A$ ="B79", atunci LEN(A$)=3

STR$ (expresie) – calculează valoarea expresiei, iar rezultatul formează un şir de caractere, ca şi cum ar fi tipărit cu instrucţiunea PRINT.

De exemplu STR$ (127.5-B)="7.5", dacă variabila B are valoarea 12∅.

CHR$ (expresie) — calculează (determină) caracterul care are codul ASCII egal cu valoarea expresiei. De exemplu CHRS(65)="A"

INKEY$ — citește un caracter de la tastatură (în cazul în care a fost acționată o tastă) și întoarce codul ASCII al caracterului. În cazul cînd nu s-a acționat nici o tastă rezultatul este șirul nul (" ").

Funcțiile VAL și LEN pot fi folosite în orice expresie (aritmetică), însă pe prima poziție în cadrul expresiei. Funcțiile STR$, CHR$ și INKEY$ pot fi folosite numai în expresii de tip șir.

Pentru executarea unei operații de intrare/ieșire, pe un port specificat, se pot folosi funcțiile :

GET(x) — citește un octet de la portul numărul x, Ø ⩽ x ⩽ 255. Valoarea funcției va fi un număr întreg, Ø ⩽ GET(x) ⩽ 255. Funcția GET poate fi folosită în orice expresie.

PUT(x) — poate apărea numai în membrul stîng al unei instrucțiuni de atribuire. Execuția funcției constă în transmiterea la portul cu numărul x, a valorii expresiei din membrul drept, al instrucțiunii de atribuire.

Exemple :

```
1Ø PUT(127)=A+17
```

Expresia din membrul drept este evaluată, convertită în întreg, iar cei mai puțin semnificativi opt biți sînt trimiși la portul 127.

Instrucțiunea : 15 PUT (127)=GET (255), va citi un octet de la portul 255 și-l va transmite la portul 127.

9.2.5. **Expresii.** Expresiile utilizate în instrucțiunile BASIC sînt de două tipuri : expresii aritmetice (le vom numi, pe scurt, expresii) și expresii șir (sau șiruri).

Expresiile șir pot conține ca operanzi : constante șir, variabile șir sau subșiruri precum și funcțiile ce au ca valoare șiruri de caractere : STR$, CHR$, și INKEY$. Ca operator, în formarea expresiilor șir, poate fi utilizat operatorul de concatenare, notat +. De exemplu : "123"+"ABC"="123ABC", sau CHR$ (65)+STR$(12+7)+"CASA"="A12CASA"

Expresiile aritmetice pot fi compuse din constante, variabile simple sau indexate, funcții, legate între ele prin operatori aritmetici etc.

Exemple : 1ØØ

```
VAL ("7Ø.5+B")
(5—x4 ↑ 3) * (Z (k-1, i)-A/LOG (Z+1))
```

Ordinea de execuție a operațiilor.

— În expresiile fără paranteze succesiunea este următoarea :

1. calculul valorilor funcțiilor,
2. ridicarea la putere,
3. înmulțiri și împărțiri (cu prioritate egală),
4. adunări și scăderi (cu prioritate egală).

— Dacă două sau mai multe operații de aceeași prioritate apar într-o expresie, ele se efectuează în ordine, de la stînga la dreapta, cu excepția ridicării la putere, unde execuția este de la dreapta spre stînga.

Exemplu:

3 ↑ 2 ↑ 3=3 ↑ 8=6561

— Dacă într-o expresie apar paranteze, se vor executa întîi operațiile cuprinse între parantezele interioare.

Observații:

1. Singurele paranteze admise sînt parantezele rotunde.
2. Operațiile aritmetice trebuie scrise explicit.

Exemplu:

54B se va scrie: 5*A*B.

9.2.6. **Instrucțiuni și comenzi.** Elementele principale ale limbajului BASIC sînt instrucțiunile și comenzile limbajului. Sintactic, instrucțiunile se deosebesc de comenzi prin faptul că orice instrucțiune este etichetată, adică orice instrucțiune începe cu un număr întreg (pe care-l vom numi număr de linie), cuprins între 0 și 32767.

Numerele de linie au un dublu rol:

— determină ordinea de execuție a instrucțiunilor (instrucțiunile pot fi introduse în orice ordine, însă vor fi executate în ordinea crescătoare a numerelor de linie);

— sînt utilizate în instrucțiunile de transfer, pentru referirea instrucțiunilor.

Comenzile și instrucțiunile vor fi prezentate detaliat în capitolele următoare.

9.2.7. **Exerciții**

1. Dacă dispuneți de un tabel cu logaritmi, încercați să verificați următoarea regulă: a ridica 10 la o putere este echivalent cu inversul logaritmului din acel număr. Introduceți:

```
10 PRINT 10↑0.3020
20 END
RUN
```

și urmăriți în tabela de logaritmi, inversul logaritmului din 0.3020. De ce cele două rezultate nu sînt exact egale?

Observație: În continuare, instrucțiunea END de la sfîrșitul programului și comanda RUN ce lansează în execuție programul, nu vor mai fi specificate explicit, considerîndu-se subînțeles.

2. S-a văzut că numerele sînt reprezentate în formatul cu virgulă mobilă, format care permite reprezentarea unei game largi de valori numerice reale, precizia reprezentării (6—7 cifre semnificative) nedepinzînd de valoare. Executați însă următoarea instrucțiune:

```
10 PRINT 1E8+1—1E8, 1E8—1E8+1
```

Observați că pentru calculator 1E8 și 1E8+1 sînt valori egale.

3. Pentru obținerea logaritmilor zecimali (cei găsiți în tabelele de logaritmi), împărțiți logaritmul natural prin LOG (10). Astfel, pentru a calcula log 2 se folosește instrucțiunea:

```
10 PRINT LOG (2)/LOG (10)
```

obţinînd rezultatul ∅.3∅1∅29. Încercaţi să faceţi înmulţiri şi împărţiri cu logaritmi servindu-vă de calculator ca de o tabelă de logaritmi. Pentru test, puteţi folosi exerciţiul 1, care calculează inversul logaritmului.

4. EXP şi LOG sînt funcţii inverse una alteia. Aplicîndu-le succesiv unui număr, îl vor lăsa neschimbat.

Exemplu : LOG (EXP (2))=EXP (LOG (2))=2.

Acest lucru este valabil şi pentru funcţiile TAN şi ATN. Puteţi utiliza acest exerciţiu pentru a testa precizia de calcul a funcţiilor respective de către calculator.

5. Se ştie că π radiani reprezintă 18∅°. Pentru a transforma o mărime exprimată în grade, în radiani, o împărţim prin 18∅ şi înmulţim cu π, astfel :

```
1∅ PRINT TAN(45/180 * PI)
```

calculează tan 45∅ (=1). Pentru a transforma radianii în grade, trebuie să împărţim prin π şi să înmulţim cu 18∅.

6. Cum utilizaţi funcţiile RND şi INT pentru a obţine un număr aleator între 1 şi 6, care ar putea reprezenta aruncarea unui zar ? (Răspuns : INT (RND (1) * 6+1).

7. Pentru a testa precizia cu care este memorat π de către calculator, executaţi instrucţiunea : 1∅ PRINT PI, PI-3 PI-3.1 PI-3.14, PI-3.141.

8. Funcţia INT rotunjeşte la întreg prin lipsă.

Pentru a rotunji la întregul cel mai apropiat se adună ∅.5.

Exemple :

```
INT (2.9+∅.5)=3                    INT (5.4+∅.5)=5
INT (—2.9+∅.5)=—3                  INT (—5.4+∅.5)=—5
```

Comparaţi cele de mai sus cu rezultatele obţinute în cazul că nu adunaţi ∅.5.

9. De ce numele unei variabile trebuie să înceapă cu o literă ?

1∅. Reamintim cîteva reguli de calcul cu puteri

A ↑ ∅=1

A ↑ (—B)=1/A ↑ B

A ↑ (1/B)=rădăcina de ordin B din A

A ↑ (B+C)=A ↑ B*A ↑ C

şi

A ↑ (B * C)=(A ↑ B) ↑ C

unde A şi B sînt numere întregi pozitive.

Testaţi aceste reguli utilizînd calculatorul pentru calculul diferitelor expresii conţinînd ↑ :

De exemplu :

```
1∅ PRINT 3 ↑ (2+∅), 3 ↑ 2 * 3 ↑ ∅
2∅ PRINT 4 ↑ (—1), 1/4
```

11. Valoarea lui e, baza logaritmilor naturali, este 2.71828. Ea este memorată intern cu numele EE. Verificaţi dacă :

EXP (X)=EE ↑ X

pentru diverse valori ale lui X.

12. Executaţi următoarele instrucţiuni :

```
1∅ LET A$="ATN(1) * 4"
2∅ PRINT A$ ; "=" ; VAL (A$)
```

(Rezultatul tipărit este π)

Repetaţi execuţia schimbînd instrucţiunea 10.

De exemplu :

```
 5 LET X=9
10 LET A$="X ↑ 3+2 * X"
```

Reexecutaţi pentru expresii din ce în ce mai complicate atribuite variabilei şir A$.

13. Unele versiuni ale limbajului BASIC utilizează următoarele funcţii pentru lucrul cu şiruri : LEFT$, RIGHT$, MID$ şi TL$, cu următoarele semnificaţii :

LEFT$ (A$, N) — calculează subşirul lui A$ compus din primele N caractere.

RIGHT$ (A$, N) — calculează subşirul lui A$ compus din ultimele caractere începînd cu al N-lea.

MID$ (A$, N1, N2) — calculează subşirul lui A$ compus din N2 caractere începînd cu al N1-lea caracter din A$.

TL$ (A$) — calculează subşirul lui A$ format din toate caracterele lui A$ cu excepţia primului.

Încercaţi să descrieţi funcţiile de mai sus cu ajutorul decupării cu TO.

14. Executaţi programul :

```
1Ø A$=CHR$ (INT (RND (Ø)*1Ø+48))
2Ø PRINT A$
3Ø GOTO 1Ø
```

(instrucţiunea GOTO 1Ø indică repetarea execuţiei de la linia 1Ø).

Programul tipăreşte o cifră (între Ø÷9), deoarece 48 este codul cifrei Ø la care se adună un număr aleator între Ø şi 9, dînd codul ASCII al unei cifre (între 48 şi 57). Puteţi da altă soluţie pentru generarea unui număr întreg între Ø şi 9 ?

9.3. Comenzile şi modul de utilizare al interpretorului BASIC

9.3.1. **Lansarea în execuţie a interpretorului BASIC.** Interpretorul BASIC este depus în memoria cu conţinut permanent (PROM), începînd de la adresa 8ØØH (hexazecimal). De aceea poate fi lansat în execuţie cu ajutorul comenzii GØ8ØØ, a monitorului sistemului, ca orice program utilizator. Pentru simplitate, s-a prevăzut între comenzile monitorului, o comandă specială pentru a lansa în execuţie interpretorul BASIC, anume comanda B*.

Cînd interpretorul BASIC intră în execuţie, va tipări pe display mesajul READY, indicînd faptul că se aşteaptă instrucţiuni sau comenzi de la utilizator. În continuare, utilizatorul va introduce un program de la tastatură sau va citi un program de pe casetă magnetică. Programul astfel introdus poate fi executat, listat pe display sau modificat (cu ajutorul unor facilităţi de editare) după dorinţa utilizatorului. După execuţie, programul poate fi salvat pe casetă şi şters din memorie, pentru a se introduce un nou program.

La terminarea unei sesiuni de lucru cu interpretorul BASIC, se va ieşi de sub controlul interpretorului acţionînd comutatorul (RESET) de la consolă, lansîndu-se în execuţie monitorul sistemului.

Există două situaţii speciale de lansare în execuţie a interpretorului BASIC.

* Comanda B este disponibilă numai în versiunea VØ.1.

În prima situaţie, se utilizează subrutine în limbaj maşină, apelate dintr-un program BASIC. În acest caz, înainte de lansarea în execuţie, a interpretorului, se va introduce în memorie codul pentru subrutinele apelate, cu ajutorul comenzilor monitorului. Apoi se lansează interpretorul BASIC de la adresa 815H (deci cu comanda GØ815).

În cea de-a doua situaţie lansarea interpretorului are loc de la adresa 829H, fără iniţializări. Lansarea de la această adresă (cu GØ829) se foloseşte cînd în memoria utilizată de interpretor pentru păstrarea programelor, se află un program care nu trebuie şters (de exemplu cînd s-a ieşit din BASIC în Monitor şi se doreşte să se intre iarăşi sub controlul interpretorului, fără ştergerea programului BASIC introdus în sesiunea anterioară).

9.3.2. **Editarea programului.** Există facilităţi de corectare a unei linii în cursul introducerii ei de la consolă (tastatură), sau de editare a programului deja introdus (ştergerea unei linii sau înlocuirea ei cu altă linie).

Astfel :

— ştergerea ultimului caracter introdus se realizează acţionînd DEL

— ştergerea liniei în curs de introducere se face acţionînd simultan CTRL şi Y

— ştergerea unei instrucţiuni din program se face prin tipărirea numărului ei de linie şi RETURN.

O instrucţiunea se poate înlocui prin tipărirea noii instrucţiuni cu acelaşi număr de linie cu vechea instrucţiune.

Observaţie : *Orice linie introdusă de la consolă (instrucţiune, comandă sau linie de date) va fi luată în considerare de sistem la acţionarea lui RETURN.*

9.3.3. **Listarea şi salvarea pe casetă a unui program.** Un program BASIC aflat în memorie poate fi listat la display sau salvat (stocat) pe casetă magnetică. Aceste operaţii pot fi executate indiferent dacă programul a fost executat sau nu. Pentru listarea programului se foloseşte comanda :

LIST N1,N2

unde : N1 şi N2 sînt numere de linie.

Execuţia comenzii constă în listarea programului existent în memorie în ordinea crescătoare a numerelor de linie. Parametrii N1 şi N2 sînt opţionali. În cazul în care se specifică un singur număr de linie se vor lista instrucţiunile ce au numărul de linie mai mare sau egal cu numărul specificat în comanda LIST. Cînd se specifică ambii parametrii, se vor lista instrucţiunile care au numărul de linie cuprins între N1 şi N2 inclusiv. Execuţia comenzii poate fi oprită de la consolă, acţionînd CTRL.

Pentru salvarea pe casetă a unui program se foloseşte comanda SAVE. Dacă programul foloseşte subrutine în limbaj maşină, la comanda SAVE se salvează automat şi subrutinele apelate. Dacă comanda SAVE este executată după ce programul a fost executat, atunci se salvează pe casetă şi valorile variabilelor utilizate în execuţia programului.

Deoarece identificarea programelor de pe casetă se face manual de către utilizator, este bine ca înainte de salvarea unui program să se înregistreze

pe casetă un text vorbit (titlu), referitor la programul ce urmează, necesar identificării. Apoi se conectează casetofonul pe înregistrare la calculator și se lansează în execuție comanda SAVE.

9.3.4. **Citirea unui program de pe casetă.** Pentru citirea unui program de pe casetă se folosește comanda LOAD. Înainte de a executa comanda LOAD, trebuie să se depisteze (sonor) începutul programului (după titlul înregistrat la salvarea programului) și apoi să se conecteze casetofonul la calculator, pe redare. Programul este citit de pe casetă în memoria calculatorului împreună cu eventualele subrutine în limbaj mașină și cu variabilele, exact în starea în care a fost salvat, vechiul program din memorie fiind șters.

Pentru o citire cît mai fiabilă este bine ca volumul redării să fie reglat corespunzător.

9.3.5 **Execuția unui program.** Pentru a lansa în execuție un program BASIC aflat în memorie se pot folosi comenzile :

RUN nr. linie

sau

GOTO nr. linie

execuția programului începînd de la linia cu numărul specificat în comandă. Pentru comanda RUN, numărul de linie poate să lipsească, caz în care execuția începe cu prima instrucțiune din program. Deosebirea dintre cele două comenzi constă în faptul, că RUN inițializează (șterge) variabilele înainte de lansarea în execuție, pe cînd GOTO păstrează valorile variabilelor obținute într-o execuție anterioară. De exemplu, dacă se citește un program de pe casetă împreună cu valorile variabilelor (eventual tablouri), atunci lansarea sa în execuție se va face cu comanda GOTO n1, n1 fiind numărul de linie al instrucțiunii aflată după instrucțiunea DIM în program, pentru a se putea folosi valorile variabilelor tablou salvate pe casetă. Dacă nu se dorește utilizarea valorilor variabilelor, obținute într-o execuție anterioară, programul poate fi lansat în execuție cu comanda RUN.

Observație : Pentru a nu se salva valorile variabilelor pe casetă atunci cînd nu sînt necesare, trebuie să se știe faptul că tabela de variabile este ștearsă în următoarele cazuri :

— la introducerea unei noi linii în program,
— la ștergerea unei linii din program,
— la execuția comenzii SCRATCH (însă în acest caz este șters și programul).

Execuția unui program este oprită la :

— depistarea unei erori în program,
— execuția unei instrucțiuni STOP sau END
— întreruperea de la consolă, acționînd caracterul CTRL,
— întreruperea de la consolă, acționînd RESET, caz în care se intră sub controlul Monitorului.

La terminarea execuției, sistemul tipărește READY. După eventuale corectări inserări de instrucțiuni, programul poate fi executat din nou.

9.3.6. **Ştergerea unui program din memorie.** Înainte de introducerea unui nou program de la tastatură, vechiul program din memorie trebuie sters, altfel liniile lui vor interfera cu liniile noului program. Pentru ştergerea programului din memorie (inclusiv a eventualelor subrutine în limbaj maşină) se foloseşte comanda : SCRATH (sînt suficiente primele trei caractere : SCR).

9.3.7. **Exerciţii**

1. S-a văzut că variabilele utilizate de un program în timpul execuţiei sînt şterse din memorie la introducerea sau ştergerea unei instrucţiuni a programului şi la comanda SCR. În toate aceste cazuri, este alterat însă şi programul o dată cu ştergerea variabilelor (desigur, există posibilitatea de a şterge o linie şi a o reintroduce cu acelaşi conţinut).

O posibilitate de a şterge variabilele fără a altera programul, este prin execuţia comenzii RUN n1, unde n1 este numărul de linie al instrucţiunii END sau STOP din program.

Acest lucru este valabil deoarece, la începutul execuţiei programului, prin comanda RUN se şterg variabilele utilizate de program (într-o execuţie anterioară), iar apoi se va executa instrucţiunea END sau STOP. Ştergerea variabilelor este utilă, în cazul salvării pe casetă a programelor ce conţin variabile (eventual tablouri), ale căror valori nu sînt necesare la reîncărcarea de pe casetă. Testaţi acest lucru, pentru programe care conţin tablouri suficient de mari, şi veţi observa diferenţa de timp la salvare şi încărcare pe casetă.

2. Executaţi următorul program :

```
10 LET X=7
20 PRINT X
30 END
```

folosind comanda RUN. Executaţi acum programul începînd de la linia 20 cu comenzile GOTO20 şi RUN20. Observaţi că rezultatul este diferit (în primul caz este 7 în al doilea 0). Acest lucru se datorează faptului că, comanda RUN şterge variabilele rămase dintr-o execuţie anterioară (lucru care s-a mai spus), iar interpretorul BASIC consideră variabilele nedefinite ca avînd valoarea zero.

3. Pentru a testa faptul că comanda SAVE salvează împreună cu programul şi variabilele utilizate, se poate introduce un program, de exemplu cel de la exerciţiul 2, şi după execuţie se va salva pe casetă. Pentru salvare se vor executa următoarele acţiuni :

— poziţionarea benzii în zona în care se doreşte să se înregistreze programul,
— utilizînd microfonul, se va da un nume programului ; acest lucru nu este absolut necesar, dar este util pentru regăsirea ulterioară a programului,
— se conectează casetofonul (prin cablul corespunzător) cu calculatorul,
— se introduce comanda SAVE (fără a acţiona tasta RETURN),
— se comută casetofonul pe înregistrare,
— se acţionează RETURN,
— se urmăreşte apariţia pe ecranul televizorului a mesajului de sfîrşit de operaţie (READY). La apariţia mesajului, opriţi casetofonul. Pe durata înregistrării se va auzi un zgomot specific, în difuzorul asociat calculatorului, iar imaginea pe televizor va fi formată din dungi variabile (în primele 10 secunde din înregistrare, se depune un preambul necesar sincronizării, iar apoi programul BASIC şi variabilele).

În continuare, se şterge programul din memorie cu ajutorul comenzii SCR (se poate testa, că memoria nu conţine nici un program, introducînd comanda LIST).

În continuare, se încarcă programul de pe casetă în memorie executînd următoarele acţiuni :

— se rebobinează banda pentru a o aduce la începutul fişierului (zonei) cu programul. Pentru depistarea începutului programului se va face uz de numele înregistrat înaintea programului,
— se conectează casetofonul la calculator, pe redare,
— se reglează volumul redării suficient de mare (3/4), iar tonalitatea la valoare înaltă (dacă casetofonul are reglaj de tonalitate),
— se introduce comanda LOAD (fără a acţiona tasta RETURN),

— se porneşte casetofonul,

— se urmăreşte dacă beculeţul de la consolă s-a aprins şi apoi se acţionează RETURN,

— se urmăreşte apariţia pe ecranul televizorului a mesajului de sfîrşit de încărcare. Astfel, în cazul în care operaţia a decurs fără erori, vor fi afişate două numere în hexazecimal care reprezintă : adresa de început a zonei de memorie salvată pe casetă şi respectiv — lungimea ei (numărul de octeţi). Apoi este tipărit mesajul READY, de către interpretor. În cazul în care s-au depistat erori la încărcare, monitorul va tipări ? şi va prelua controlul. În acest caz, încărcarea programului de pe casetă trebuie reluată, însă este indicat a se intra în BASIC din monitor cu comanda G∅829 (fără iniţializări), pentru a se lista (cu LIST) programul încărcat şi a se constata amploarea erorii de încărcare. Uneori eroarea poate fi corectată editînd o anumită linie, fără a mai încărca încă odată programul de pe casetă.

După încărcarea programului, se execută comanda GOTO2∅, şi se observă că rezultatul este 7, deci împreună cu programul a fost salvată şi variabila X.

4. Introduceţi un program BASIC şi apoi opriţi alimentarea de la reţea a calculatorului. Realimentînd calculatorul şi lansînd interpretorul BASIC veţi constata că programul s-a pierdut. Deci, în cazul programelor mari, este indicat salvarea lor pe casetă, chiar în faza de punere la punct, pentru a nu risca reintroducerea lor în întregime.

9.4. Instrucţiunile limbajului BASIC

9.4.1. **Exemplu de program.** Înainte de a începe prezentarea instrucţiunilor limbajului BASIC, se va da un exemplu de program scris în BASIC, pentru rezolvarea sistemului de două ecuaţii lineare cu două necunoscute.

$$ax+by=c$$

$$dx+ey=f$$

Sistemul are soluţie unică dacă ae-bd≠∅, caz în care :

$$x=\frac{ce-bf}{ae-bd} \quad \text{şi} \quad y=\frac{af-cd}{ae-bd}$$

Dacă ae—bd=∅, atunci, fie nu există nici o soluţie (sistem imposibil), fie există o infinitate de soluţii (sistem nedeterminat) însă, nu există soluţie unică. Programul este următorul :

```
1∅ REM "PROGRAM CE REZOLVA UN SISTEM"
15 REM "DE DOUA ECUATII LINEARE"
2∅ PRINT "INTRODUCETI COEFICIENTII A, B, C, D"
3∅ INPUT A, B, C, D
4∅ LET G=A*E—B*D
5∅ IF G=∅ THEN 12∅
6∅ PRINT "INTRODUCETI TERMENII LIBERI C, F"
7∅ INPUT C, F
8∅ LET X=(C*E—B*F)/G
9∅ LET Y=(A*F—C*D)/G
1∅∅ PRINT X, Y
11∅ STOP
12∅ PRINT "SISTEMUL NU ARE SOLUTIE UNICA"
13∅ END
```

Privind acest program se observă în primul rînd că, în scrierea sa, sînt folosite numai litere mari. În al doilea rînd, se constată că fiecare linie a programului începe cu un număr ; el este numărul de linie discutat în capitolul precedent. În al treilea rînd, se observă că fiecare instrucţiune începe, după

numărul de linie, cu un cuvînt care determină tipul instrucţiunii. Astfel, în exemplul dat sînt folosite şapte instrucţiuni (REM, PRINT, INPUT, LET, IF, STOP şi END).

O altă observaţie, care nu este evidentă din program, constă în aceea că, în BASIC spaţiile nu au nici o semnificaţie, cu excepţia celor din textele cuprinse între semnele apostrof (ca în liniile 1Ø,2Ø,6Ø).

În continuare vor fi prezentate, pe rînd, instrucţiunile limbajului BASIC

9.4.2. **Comentarea unui program.** Primele două instrucţiuni, din exemplul de mai sus, nu au nici un rol în rezolvarea sistemului de ecuaţii. Ele sînt scrise pentru a introduce comentarii într-un program BASIC.

Formatul instrucţiunii este :

nr. linie REM comentariu

Instrucţiunea REM poate apărea oriunde se doreşte introducerea unui comentariu, în cadrul unui program. *Deoarece sistemul elimină spaţiile din orice şir de caractere neinclus între semnele apostrof (") este bine ca un comentariu să se includă între aceste semne.*

Exemplu :

```
13Ø REM "COMENTARIILE OCUPA MEMORIE UTILA"
```

9.4.3. **Terminarea unui program.** Pentru a opri execuţia unui program, pot fi utilizate instrucţiunile END sau STOP.

Formatul instrucţiunii STOP este :

nr. linie STOP

La întîlnirea instrucţiunii STOP în program, sistemul va scrie mesajul :

STOP AT NN

unde NN este număru de linie al instrucţiunii STOP care a produs oprirea execuţiei. Într-un program pot fi utilizate mai multe instrucţiuni STOP, în funcţie de necesităţi. În exemplul din paragraful 9.4.1. instrucţiunea STOP are numărul de linie 11Ø.

Formatul instrucţiunii END este :

nr. linie END

Spre deosebire de instrucţiunea STOP, într-un program trebuie să existe o singură instrucţiune END şi să fie ultima instrucţiune din program (să aibă numărul de linie cel mai mare).

9.4.4. **Instrucţiunea de atribuire (LET).** Formatul instrucţiunii LET este următorul :

nr. linie variabilă = expresie

sau

nr. linie LET variabilă = expresie

unde :

— variabila poate fi o variabilă numerică sau variabilă şir, simplă sau indexată, sau funcţia PUT ;

— expresia din membrul drept poate fi aritmetică sau expresie şir.

Execuţia instrucţiunii constă în evaluarea expresiei din membrul drept şi atribuirea valorii obţinute, variabilei din membrul stîng.

Exemple:

```
5 B=B+1                  2Ø C$=INKEY$
1Ø A$="ABCD179"          3Ø LET A=4.17+C
```

Observaţie:

O variabilă utilizată într-o expresie, fără a fi fost definită în prealabil, primeşte automat valoarea zero. Astfel, în instrucţiunea 3Ø de mai sus, dacă variabila C nu a fost definită, valoarea lui A după execuţia instrucţiunii 3Ø va fig4.17.

```
5Ø X=X+Y ↑ 3.5
6Ø LET W7=((W—X)/4.3)*SQR (Z(I)—A)/B
9Ø A$ (7 TO 1Ø)=CHR$ (65)+„123"
```

Dacă expresia şir atribuită unei variabile şir, este mai lungă decît dimensiunea variabilei (stabilită prin DIM sau printr-o atribuire anterioară), atunci se atribuie variabilei şir doar prima parte din expresia şir (egală cu lungimea variabilei), restul expresiei şir ignorîndu-se. Dacă lungimea şirului din membrul drept este mai mică decît lungimea variabilei şir, restul de caractere se completează cu spaţii.

Exemple:

```
1Ø A$="12345678"
5Ø A$ (2TO5)=„ABCDEFG"
```

În acest caz, A $ va deveni “1ABCD678”, deci, au fost păstrate doar primele patru caractere din şirul "ABCDEFG". Dacă în locul instrucţiunii 2Ø de mai sus, se folosea instrucţiunea:

```
2Ø A$ (2TO5)="AB"
```

valoarea variabilei A $ ar fi devenit "1AB 678"

```
3Ø B$=STR$ (A+B—17)+"MEDIA"
4Ø J (I, INT (K ↑ 1Ø))=COS (EXP (K+1))
```

9.4.5. Utilizarea variabilelor indexate (DIM)

Formatul instrucţiunii este:

nr. linie DIM tablou 1 (dimens), tablou 2 (dimens, dimens) ...

unde:

— tablou 1, tablou 2, sînt identificatori (nume) de tablou (vectori, matrici sau variabile şir).

— dimens reprezintă expresii ale căror valori definesc dimensiunile tablourilor; este necesară îndeplinirea condiţiei:

$1 \leqslant dimens \leqslant 254$

Tablourile numerice pot avea una sau două dimensiuni (vectori sau matrici)

Exemplu :

```
110 DIM A (50), B (M, N)
```

Instrucţiunea 110 declară două tablouri A şi B, A fiind un vector cu 50 de elemente, iar B o matrice cu M×N elemente (M, N cunoscute).

Execuţia instrucţiunii constă în alocarea, memoriei pentru tablourile declarate şi iniţializarea tuturor elementelor ale fiecărui tablou numeric cu zero.

Tablourile de variabile şir trebuie declarate în instrucţiunea DIM, indicîndu-se numărul de variabile şi lungimea variabilelor.

Exemplu :

```
10 DIM A$ (15, 20), B$ (50)
```

Instrucţiunea 10 declară un tablou A $ format din 15 variabile şir a cîte 20 de caractere fiecare şi o variabilă şir B $ cu lungimea de 50 de caractere. *De remarcat faptul că, variabilele şir (exemplu B $) nu trebuie declarate într-o instrucţiune DIM, ele putînd fi utilizate direct în instrucţiuni de atribuire sau de intrare/ieşire.* Tablourile şir trebuie declarate într-o instrucţiune DIM înainte de utilizare. Folosirea lor se face prin specificarea unei anumite variabile şir componente, şi nu a întregului tablou. De exemplu, instrucţiunea :

```
20 A $ ="1234"
```

este incorectă pentru că, A $ a fost declarat tablou în instrucţiunea 10.

Corect ar fi :

```
20 A$ (2)="1234"
```

instrucţiunea în urma căreia, variabila a doua din tabloul A $ primeşte valoarea "1234". Tot corectă este şi instrucţiunea :

```
20 A$ (2,10 TO 13)="1234"
```

În cazul tablourilor (variabilelor) şir, execuţia instrucţiunii DIM constă în alocarea de memorie, cîte un octet pentru fiecare element (caracter), iniţializat cu codul spaţiului (A0H).

Tablourile numerice şi cele şir pot fi declarate intercalat, în aceeaşi instrucţiune DIM.

Exemplu :

```
10 DIM A (10, 10), B$ (100), C (10)
```

9.4.6. Exerciţii

1. Introduceţi următoarea secvenţă de instrucţiuni :

```
10 LET A$="*X*+*Y"          30 LET A$ (4)=CHR$ (162)
20 LET A$ (2)=CHR$ (162)    40 PRINT A$
```

(162 este codul ghilimelelor)

De remarcat că secvența de mai sus nu este echivalentă cu :

```
10 LET A$="X" "+" "Y"
```

deoarece, în acest caz, membrul drept al instrucțiunii de atribuire este tratat ca o expresie de tip șir, evaluată la "XY". O altă modalitate corectă de a introduce ghilimele într-un șir este introducerea sucesivă. Actfel instrucțiunea :

```
10 LET A$="X" "+" "Y"
```

este echivalentă cu secvența de mai sus.

2. Fie secvența de instrucțiuni :

```
10 LET A$="12345678"      40 PRINT A$
20 PRINT A$               50 LET A$="1234"
30 LET A$="ABC"           60 PRINT A$
```

După execuție se va obține eroarea 28 în linia 50. Acest lucru este consecința faptului că în linia 30 are loc o redimensionare a șirului A$, la lungimea de 3 caractere (de la 8 cite avea inițial). Pentru a evita această redimensionare, linia 30 se poate scrie 30 LET A$ (TO)="ABC", șirul A$ rămînînd de lungime 8, prin completarea șirului "ABC" cu 5 spații. Deci, trebuie reținut faptul că în gestiunea șirurilor nu se folosesc doi indicatori de lungime : unul păstrînd lungimea maximă, iar celălalt lungimea efectivă a șirului (sau lungimea maximă să fie pentru toate șirurile o constantă, exemplu 256 caractere).

3. Introduceți programul :

```
5 DIM B$ (3, 8)               40 PRINT B$ (2)
10 LET B$ (2)="12345678"      50 LET B$ (2)="12345"
20 PRINT B$ (2)               60 PRINT B$ (2)
30 LET B$ (2)="MPQ"           70 END
```

Executînd programul nu se va obține eroare, ca în cazul exercițiului precedent. Deci, în cazul tablourilor de șiruri, nu are loc redimensionarea unui șir component la o valoare mai mică, ci se păstrează dimensiunea (8) declarată în DIM, prin completarea automată cu spații. Pentru a nu se completa cu spații (păstrîndu-se totuși dimensiunea de 8 caractere) se poate utiliza specificarea cu TO. Astfel instrucțiunea 30 devine : 30 LET B$ (2,1 TO3)="MPQ".

În acest caz, șirul tipărit de instrucțiunea 40 va fi : "MPQ45678".

4. În cazul utilizării unor tablouri mari se poate obține la execuție mesajul de eroare MEMORY FULL, indicînd faptul că nu mai există memorie suficientă pentru alocarea tabloului. Trebuie să se aibă în vedere faptul că, pentru fiecare element al unei variabile tablou numerice, se alocă cîte 4 octeți. Astfel, instrucțiunea 10 DIM A (11,11), B (16,16) va aloca două tablouri numerice, primul ocupînd aproximativ 0,5 Ko și al doilea 1 Ko. Ce acțiuni se pot indica la apariția acestui mesaj de eroare ?

9.4.7. **Instrucțiuni de intrare/ieșire.** Instrucțiunile de intrare/ieșire permit să se introducă date (valori ale variabilelor) pentru program și să tipărească rezultate sau mesaje către utilizator. În execuția acestor instrucțiuni sînt folosite în mod obișnuit tastatura și display-ul, însă pot fi utilizate oricare din perifericele sistemului, dacă se au în vedere facilitățile mai deosebite de intrare/ieșire materializate prin funcțiile GET și PUT sau subrutine în limbaj mașină, apelate din programul BASIC, cu instrucțiunea CALL.

Instrucțiunea INPUT.

Format:

nr. linie INPUT listă de variabile

Lista de variabile poate conţine variabile simple şi variabile indexate sau variabile şir, separate prin virgule.

Instrucţiunea INPUT este folosită pentru atribuirea de valori variabilelor din listă, valorile fiind introduse de la tastatură (consolă) în timpul execuţiei programului.

Atunci cînd se execută instrucţiunea INPUT, sistemul va tipări la consolă(:) indicînd că aşteaptă date de la consolă. Se vor introduce constantele dorite separate prin virgule, linia introdusă va fi luată în considerare de sistem la acţionarea tastei RETURN. Dacă s-au introdus mai puţine date decît variabile în instrucţiunea INPUT, sistemul va tipări din nou (:) şi introducerea datelor va continua pînă ce toate variabilele au primit valori.

Dacă se introduc mai multe constante decît sînt necesare, constantele suplimentare se ignoră. În cazul în care se doreşte oprirea execuţiei programului, în timp ce sistemul aşteaptă date de la consolă, se acţionează tasta CTRL şi C simultan, iar sistemul va tipări READY.

Exemplu :

```
50 INPUT I1, A, B, V (2).
```

La execuţia instrucţiunii 50 sistemul va tipări (:) la consolă şi va aştepta introducerea a patru constante. Dacă se introduc doar trei numere, sistemul va tipări iarăşi (:) indicînd că aşteaptă date. Astfel, la consolă va apărea:

```
: 4, 17.5, 9E10
: 10.3
```

Dacă este comisă o eroare în timpul introducerii de date, sistemul va tipări un mesaj de eroare, iar ultima linie de date va trebui reintrodusă. Astfel, dacă în exemplul de mai sus s-ar fi introdus:

```
: 4, 17M5, 9E10, 10.3
```

sistemul ar fi răspuns:

```
INPUT ERROR, TRY AGAIN
: (aşteaptă noi date)
```

Introducerea datelor se repetă corect astfel:

```
: 4, 17.5, 9E10, 10.3
```

În cazul folosirii variabilelor şir, trebuie remarcat că, din linia cu constante şir corespunzătoare, introdusă de la consolă, sînt eliminate spaţiile neincluse între ghilimele.

Exemple :

```
10 INPUT A$, B, C$ (2TO5)
: "SINT GATA", 73, ABCD
```

Variabila A $ a primit valoarea "SINT GATA", variabila B valoarea 73 iar variabila C$ valoarea ABCD, pentru elementele 2 la 5. Textul SINT GATA a fost introdus între ghilimele, pentru a se păstra spaţiul dintre cuvinte

Instrucţiunile READ şi DATA. Instrucţiunea READ are acelaşi efect cu INPUT, cu deosebirea că datele nu sînt introduse de la consolă, ci sînt citite dintr-un bloc de date, definit cu instrucţiunea DATA.

Formatul instrucţiunilor :

nr. linie READ listă de variabile

nr. linie DATA listă de constante.

Pentru a putea ţine evidenţa constantelor citite in instrucţiunile DATA, interpretorul foloseşte un indicator la constanta ce urmează a fi citită din blocul de constante, care apar în instrucţiunile DATA, din program. În cazul în care nu se pot iniţializa toate variabilele din instrucţiunea READ, din cauza epuizării constantelor din instrucţiunile DATA, sistemul va da un mesaj de eroare.

O instrucţiune DATA nu este asociată unei instrucţiuni READ, ci toate instrucţiunile DATA sînt tratate ca şi cum ar forma un bloc de date.

Instrucţiunea DATA poate apărea oriunde, în cadrul programului.

Exemplu :

```
150 READ X, Y, Z
200 READ A
250 FOR I=1 TO 10
255 READ B (I)
260 NEXT I
. . . .
400 DATA 4.2, 7.5, 25, —1, .1, .01, .001, —.1
450 DATA 2., 1, 7, 9, 1E7, 3.5
```

Primele trei constante sînt citite pentru X, Y şi Z.

Valoarea —1 va fi atribuită lui A. Următoarele zece valori de la .1 la 3.5 vor fi atribuite elementelor vectorului B.

Instrucţiunea RESTORE

Format :

nr. linie RESTORE

Este folosită pentru a iniţializa indicatorul din blocul DATA, pe prima instrucţiune DATA din program. Astfel, se permite reutilizarea datelor.

Exemplu :

```
20 FOR K=0 TO 10
30 READ B (K)
40 NEXT K
50 RESTORE
60 READ X, Y, Z
70 RESTORE
. . . .
200 READ
500 DATA 1, 2, 3, 4, 5, 6, 7, 8, 9, 10
```

Elementele vectorului B vor primi valorile 1,2,. . . , 10 ; variabilele X, Y, Z vor primi valorile 1, 2, 3, iar instrucţiunea 200READ va citi începînd cu 1.

În instrucţiunea READ pot fi folosite şi variabile şir.

Exemplu :

```
10 READ A$, B, C$
. . . .
100 DATA "ABC 123", 17.5, AB123
```

Variabila A$ primeşte valoarea "ABC 123". iar C$ valoarea AB 123.

Instrucţiunea PRINT

Formatul instrucţiunii este :

nr. linie PRINT listă
sau
nr. linie PRINT

lista poate conţine : constante, variabile simple sau indexate, expresii sau şiruri.

Instrucţiunea PRINT este folosită pentru tipărirea valorilor curente ale expresiilor care apar în listă. În cazul cînd lista lipseşte se va trece la începutul liniei următoare pe display.

Numerele reale sînt tipărite în următorul format :

— dacă numărul este negativ se tipăreşte semnul —

— dacă valoarea absolută a numărului N este cuprinsă între :

$\emptyset.1 \leqslant N \leqslant 9999999$,

atunci numărul este tipărit fie ca număr întreg (dacă valoarea sa este întreagă), fie ca număr zecimal, folosindu-se punctul ca separator al părţii întregi de partea zecimală.

— dacă valoarea numărului nu se încadrează în intervalul de mai sus, atunci numărul este tipărit în notaţie ştiinţifică (exponenţială) astfel :

[—] X.XXXXX E [—] YY

unde : [ ] — indică partea opţională a reprezentării, dacă numărul este negativ se tipăreşte — ;
X — reprezintă o cifră a numărului ;
E — indică reprezentarea exponenţială ;
Y — este cifră a exponentului.

Valoarea numărului va fi : $[-]\ X.XXXXX \times 1\emptyset^{[-]YY}$

În funcţie de separatorul folosit între elementele din listă, se va face spaţierea între valorile tipărite.

Ca separatori pot fi folosiţi „ , " sau „ ; "

Linia pe care se tipăreşte este împărţită în zone. Lungimea unei zone este de 15 caractere. Pentru un display cu lungimea liniei de 3∅ caractere vor exista două zone începînd cu poziţiile ∅ şi 15.

Exemple :

```
1∅ LET X=5
5∅ PRINT X, (X*2) ↑ 6, X*2
6∅ PRINT X ↑ 4, X—25, X—100
8∅ END
```

La display va apărea :

```
Ø...        15......        3Ø ← poziţii
 5            1.ØØØØØE 06
 1Ø           625
−20          −95
```

Cînd valoarea de tipărit depăşeşte lungimea unei zone, elementul următor este tipărit la începutul primei zone libere.

```
1Ø LET X=25
2Ø PRINT "SQR(X)=", SQR(X)
3Ø END
```

Se va tipări :

```
Ø...        15...        3Ø ← poziţii
SQR(X)=     5
```

Folosirea ca separator a lui „;" conduce la tipărirea fără salt la următoarea zonă.

Dacă lista din instrucţiunea PRINT nu se termină cu „," sau „;" atunci, la sfîrşitul execuţiei instrucţiunii (după tipărirea tuturor elementelor din listă), se va trece la începutul liniei următoare. Terminarea listei din instrucţiunea PRINT, cu unul din separatorii „," sau „;" este utilă în cazul tipăririi mai multor valori într-un ciclu, pentru a se face economie de spaţiu de afişare.

Exemplu : Programul

```
1Ø FOR I=1 TO 1Ø
2Ø PRINT I
3Ø NEXT I
4Ø END
```

Va tipări fiecare valoare la început de linie

```
1
2
3
. . .
1Ø
```

Dacă se foloseşte separatorul „,"

```
1Ø FOR I=1 TO 1Ø
2Ø PRINT I,
3Ø NEXT I
4Ø END
```

atunci se vor tipări cîte două valori pe linie. Pentru a se putea tipări oriunde pe suprafaţa display-ului formată din 32 de linii a 3Ø de caractere (coloane) fiecare, se poate folosi în instrucţiunea PRINT funcţia : AT(linie, coloană).

Exemplu :

```
1Ø LET A$="A="
2Ø PRINT AT (16,15); A$; 1ØØ
```

va tipări la mijlocul ecranului textul A=1ØØ.

Funcţia AT este utilă în afişarea de valori sau texte, pe un desen realizat cu instrucţiunile grafice, sau pentru tipărirea într-o anumită zonă a ecranului, indiferent de poziţia curentă, în care urma să se tipărească. Poziţia (1, 1) este în colţul din stînga sus al ecranului, iar (32, 3Ø) în dreapta, jos.

9.4.8. Exerciţii

1. Instrucţiunea INPUT este deseori precedată de instrucţiunea PRINT, de exemplu:

```
1Ø PRINT "INTRODUCETI DATELE"
2Ø PRINT "X=";
3Ø INPUT X
. . .
```

Valoarea lui X va fi afişată pe acelaşi rînd cu mesajul X=, deoarece instrucţiunea 2Ø PRINT se termină cu ;.

Similar, pentru introducerea de texte:

```
1Ø PRINT "DORITI O NOUĂ EXECUTIE";
2Ø INPUT R$
. . .
```

răspunsul introdus fiind DA sau NU, în urma căruia în program se va executa o decizie.

2. Executaţi următorul program:

```
1Ø INPUT A$
2Ø PRINT A$; "="; VAL (A$)
3Ø GOTO1Ø
```

Introduceţi instrucţiuni PRINT suplimentare astfel încît calculatorul să precizeze ce intenţionează să facă şi să ceară politicos datele de intrare. De remarcat că expresia introdusă spre evaluare (în A$), poate conţine orice fel de prelucrări numerice inclusiv variabile, a căror valoare trebuie introdusă în prealabil. De asemenea, şirul A$ trebuie să fie alocat într-o instrucţiune DIM, iar în instrucţiunea 1Ø să se utilizeze notaţia (TO) pentru a împiedica realocarea

```
5 DIM A$ (4Ø)
1Ø INPUT A$ (TO)
. . .
```

3. Cum se poate opri execuţia programului din exerciţiul precedent? (CTRL/C în INPUT sau CTRL între instrucţiuni, sau · · · RESET). Dacă execuţia s-a oprit între instrucţiuni, cum se poate relua fără alterarea variabilelor?

4. Scrieţi un program care calculează suma curentă a numerelor introduse.

Indicaţie: Se utilizează variabilele T pentru păstrarea totalului curent şi V pentru numărul introdus T se iniţializează cu valoarea Ø. Se citeşte un număr în V, se adună la T şi se tipăreşte T. Se repetă cu introducerea altui număr.

5. Pentru a simula mişcarea unui simbol grafic* într-o anumită direcţie se va afişa şi se va şterge simbolul respectiv, progresînd cu cîte un pas în direcţia respectivă. Se va utiliza în acest scop instrucţiunea PRINT cu funcţia AT.

```
1Ø LET X=1
2Ø PRINT AT (5, X) ; "C" ;
3Ø PRINT AT (5, X) ; " " ;
4Ø LET X=X+1
5Ø GOTO 2Ø
```

Programul de mai sus are dezavantajul că, la terminarea liniei cînd X devine >3Ø, se continuă incrementarea sa. Pentru a evita acest lucru se introduce instrucţiunea:

```
45 IF X>3Ø THEN 1Ø
```

---

* Caracterul grafic este notat cu „C"

care, reîncepe execuția de la linia 1Ø dacă X a devenit mai mare decît 3Ø. Un alt dezavantaj este timpul scurt în care simbolul apare pe ecran, datorită intervalului de timp scurt între afișare și ștergere. Pentru a mări această durată se introduce între liniile 2Ø și 3Ø un ciclu FOR fictiv :

```
22 FOR F=1 TO 1Ø
25 NEXT F
```

Acest lucru însă, are dezavantajul că micșorează viteza globală de execuție a programului. Se recomandă următoarea soluție :

```
1Ø LET X=1
2Ø LET P=X
3Ø LET X=X+1
4Ø PRINT AT (5, P) ; " " ; AT (5, X) ; "C" ;
5Ø IF X=29 THEN 1Ø
6Ø GOTO 2Ø
```

Programul folosește două variabile. X păstrează poziția curentă a caracterului, iar P poziția anterioară, care va fi ștearsă. Simbolul apărînd mereu pe ecran, crește impresia de viteză. De remarcat că, în aceeași instrucțiune PRINT poate apare de două ori specificația AT pentru poziționare. O altă soluție constă în scrierea caracterului precedat de spațiu, PRINT AT (5, X) ; "C", astfel :

```
1Ø X=1
2Ø PRINT AT (5, X) ; "C" ;
3Ø X=X+1
4Ø IF X>29 THEN 1Ø
5Ø GO TO 2Ø
```

O îmbunătățire adusă programului se referă la utilizarea instrucțiunii FOR pentru ciclare

```
1Ø FOR X=1 TO 29
2Ø PRINT AT (5, X) ; "C" ;
3Ø NEXT X
4Ø GOTO 1Ø
```

Pentru mișcarea unui pătrățel (1/4 dintr-un caracter), pot fi utilizate similar instrucțiunile PLOT și UNPLOT. În programele de mai sus, în poziția 3Ø a liniei, va rămîne caracterul fără a fi șters. Pentru a-l șterge, se introduce (în ultimul exemplu) : 35 PRINT AT (5, 3Ø) ; " " ;

Cum puteți descrie mișcarea simultană a mai multor caractere (figuri) ?

6. Funcția STR$ este foarte utilă, dar deseori neglijată. Așa cum s-a arătat, ea este utilizată pentru conversia unei valori numerice într-un șir de caractere, identic cu cel obținut prin tipărirea sa cu PRINT. Încercați următorul program :

```
1Ø PRINT 2, STR$ (2)
2Ø PRINT 1/3, STR$ (1/3)
3Ø PRINT 9E15, STR$ (9E15)
```

Se observă că, valorile afișate sînt identice. Utilizarea principală a funcției STR$ este în formatarea rezultatelor afișate. Se vor indica cîteva exemple.

a) Așezarea punctului zecimal în poziția dorită. Executați programul :

```
1Ø LET A=RND (Ø)*1ØØØØ
2Ø PRINT A
3Ø LET A=A*1Ø
4Ø GO TO 2Ø
```

Se obține următoarea secvență :

```
Ø.152587
1.52587
15.2587
152.587
1525.87
15258.7
152587
```

Valorile sînt mai ușor de comparat dacă le putem alinia pe verticală la nivelul punctului zecimal, astfel :

```
      Ø.152587
     1.52587
    15.2587
   152.587
  1525.87
 15258.7
152587
```

Pentru aceasta se modifică programul de mai sus folosind funcţia STR$.

```
5 I=6                          30 A=A*10
10 A=RND (0)*10000             35 I=I+1
20 N=LEN (STR$ (INT (A)))      40 GO TO 20
22 PRINT AT (I, 15—N) ; A
```

Variabila I este utilizată pentru a indica linia în care se tipăreşte cu funcţia AT în instrucţiunea 22 PRINT.

Expresia LEN (STR$ (INT (A))) calculează lungimea părţii întregi a numărului. Astfel, dacă A=15.2587, atunci STR$ (INT (A))="15", şir cu lungimea 2, deci tipărirea se va începe din coloana 15-2=13, încît, toate numerele vor avea punctul în coloana 15 (exceptînd cele în notaţie ştiinţifică). Execuţia programului va fi oprită de la tastatură, acţionînd CTRL.

b) Tipărirea cu un număr dorit de cifre la partea zecimală. Se vor tipări, de exemplu, numerele cu trei cifre după punctul zecimal.

```
5 I=6                          35 IF M <> N THEN 45
10 DIM B$ (15)                 40 B$ (N+1)="."
15 A=RND (0)*100               45 PRINT AT (I, 15-C) ; B$ (TO N+4)
20 M=LEN (STR$ (A))            50 A=A*10
25 N=LEN (STR$ (INT (A)))      55 I=I+1
30 B$ (TO)=STR$ (A)+"0000"     60 GO TO 20
```

Variabila I este utilizată în acelaşi scop ca la exemplul precedent (indică numărul liniei). M conţine lungimea totală a numărului, iar N numărul de cifre al părţii întregi (ca şi în exemplul precedent). În variabila B$ se „asamblează" numărul urmat de eventuale zerouri. Dacă numărul este întreg (test realizat în instrucţiunea 35 IF), se adaugă şi punctul zecimal (în locul unuia din cele patru zerouri).

c) Economisirea spaţiului de memorie şi definirea de funcţii. Deseori este utilă manipularea numerelor tratate ca şiruri de caractere, în loc de a le prelucra numeric, iar în final, evaluarea lor cu funcţia VAL. Astfel, se pot stoca numerele sub formă de şiruri, cu ajutorul funcţiei STR$:

```
LET A$=STR$ (1024)
```

apoi, se revine la valoarea numerică cu VAL : PRINT VAL (A$).

În acest caz însă, memorarea lui 1024 ca şir în variabila A$, consumă mai multă memorie decît păstrarea lui ca număr. Dacă se aplică funcţia VAL unei expresii ca "ATN (X)*4", variabila X fiind definită în prealabil, evaluarea va fi corectă, deci, o expresie care poate conţine variabile, se defineşte o singură dată în program şi poate fi evaluată pentru diverse valori ale variabilelor, cu ajutorul lui VAL (similar cu instrucţiunea DEFFN din unele versiuni BASIC). De exemplu, pentru generarea unui număr aleator între 1 şi 10 se va defini la începutul programului şirul :

```
A$="INT (RND (0)*10+1)"
```

şi de fiecare dată cînd se doreşte un număr, se va obţine prin utilizarea funcţiei VAL (A$), N=VAL (A$).

9.4.9. Instrucţiuni de control (transfer necondiţionat, condiţionat şi ciclare)

În mod normal, execuţia instrucţiunilor unui program este secvenţială. Totuşi este necesar uneori repetarea de mai multe ori a execuţiei unor instrucţiuni (fără să fie scrise repetat) sau, în funcţie de valoarea unor date, să nu se execute instrucţiunea următoare, ci să se treacă la execuţia alteia, dintr-o altă zonă a programului.

Instrucţiunile care modifică execuţia secvenţială a programului sînt instrucţiunile de control. În BASIC ele sînt următoarele :

— instrucţiunea GOTO, care realizează transferul necondiţionat (independent de date) ;

— instrucţiunile ON şi IF pentru transfer condiţionat ;
— instrucţiunea de ciclare FOR.

Instrucţiuni de transfer necondiţionat. Pentru transferul necondiţionat se foloseşte instrucţiunea GOTO.

Format : nr. linie GOTO n

Execuţia instrucţiunii constă în transferul controlului la instrucţiunea cu numărul de linie n. Saltul se poate face la orice instrucţiune din program.

Exemplu :

```
10 LET A=SQR(X↑2+Y↑2)
. . .
50 GOTO 100
. . .
70 GOTO 10
. . .
100 PRINT A
```

Instrucţiuni de transfer condiţionat. Pentru realizarea transferului condiţionat sînt disponibile două instrucţiuni : instrucţiunea IF şi instrucţiunea ON.

Formatul instrucţiunii IF este :

nr. linie IF exp1 relaţie exp2 THEN n1

unde : — exp1 şi exp2 sînt expresii numerice sau şiruri
— n1 este un număr de linie,
— relaţie poate fi :

egalitate	=
mai mare	>
mai mic	<
mai mare sau egal	>= sau =>
mai mic sau egal	<= sau =<
diferit	<> sau ><

Dacă relaţia dintre exp1 şi exp2 este adevărată se va trece la instrucţiunea cu numărul de linie n1, dacă nu, execuţia va continua cu instrucţiunea care urmează după IF.

Exemplu :

```
100 IF X+Y=SQR(X) THEN 75
```

În cazul comparaţiei şirurilor de caractere, ambele expresii trebuie să fie de tip şir. Şirurile sînt egale dacă au aceeaşi lungime şi conţin aceleaşi caractere.

Exemplu :

```
10 LET A$="ABC"
20 IF A$="ABC" THEN 100
```

După execuţia instrucţiunii 20 se va sări la execuţia instrucţiunii 100.

Un şir se consideră mai mic decît altul, dacă are lungimea mai mică sau, în cazul lungimilor egale, dacă în ordonarea lexicografică, primul şir precede pe al doilea (cifrele şi semnele speciale preced literele — conform cu codurile ASCII).

Exemplu : "ABC" este mai mic decît "BCD".

Următoarea secvenţă de program, testează dacă de la consolă s-a acţionat caracterul "A", dacă da se continuă execuţia, dacă nu, tipăreşte caracterul citit (sau şirul nul, în cazul cînd nu s-a acţionat nici o tastă) şi opreşte execuţia :

```
10 LET A$=INKEY$
20 IF A$="A"THEN 50
30 PRINT A$
40 STOP
50 . . .
```

Programul următor scrie la mijlocul ecranului textul ABC şi îl şterge la acţionarea alternativă a unor taste :

```
10 IF INKEY$=" "THEN 10
20 PRINT AT (16, 15) "ABC"
30 IF INKEY$=" " THEN 30
40 PRINT AT (16, 15) "   "
50 GOTO 10
```

Formatul instrucţiunii ON este :

nr. linie ON expresie GOTO listă

sau

nr. linie ON expresie GOSUB listă

unde listă reprezintă un şir de numere de linie.

Fie N partea întreagă a valorii expresiei specificate în instrucţiunea ON. Execuţia instrucţiunii constă în saltul la instrucţiunea cu al N-lea număr de linie din listă. În cazul lui ON-GOSUB, fiecare număr de linie, din listă reprezintă startul unei subrutine din program. Dacă N nu se încadrează între :

$1 \leqslant N \leqslant$ numărul de elemente din listă,

atunci execuţi continuă cu instrucţiunea care urmează după ON.

Exemple :

```
10 ON M-5 GOTO 100, 500, 90, 75, 550
```

M-5 trebuie să aibă o valoare între 1 şi 5 altfel, instrucţiunea nu are nici un efect.

```
20 ON X GOSUB 1000, 2000, 3000
```

În funcţie de valoarea lui X se execută una din cele trei subrutine care încep la liniile 1000, 2000 sau 3000.

Instrucţiuni de ciclare. Instrucţiunile de ciclare sînt folosite pentru execuţia repetată a unor instrucţiuni din program (numite cicluri sau bucle în program).

Pentru acest lucru se puteau utiliza instrucţiunile de transfer. Introducerea unor instrucţiuni speciale de ciclare s-a făcut în scopul simplificării muncii programatorului.

În BASIC, pentru realizarea ciclurilor, se utilizează instrucţiunile FOR şi NEXT.

Formatul instrucțiunilor:

nr. linie FOR variabilă=exp1 TO exp2 STEP exp3

. . .

(instrucțiuni ce formează bucla)

. . .

nr. linie NEXT variabilă

unde: — variabilă reprezintă o variabilă simplă care este folosită pentru controlul numărului de repetări ale buclei. Bucla constă din instrucțiunile care se găsesc între instrucțiunea FOR și instrucțiunea NEXT, cu aceeași variabilă.

— exp1, exp2 și exp3 sînt expresii.

Semnificația acestor expresii este următoarea:

exp1 — este valoarea inițială care se atribuie variabilei din instrucțiunea FOR,

exp2 — este valoarea finală a variabilei,

exp3 — este incrementul (pasul) care se adună la valoarea variabilei FOR, de fiecare dată, cînd se execută instrucțiunile din buclă.

Observație:

Este posibil ca exp3 să nu se specifice, instrucțiunea FOR avînd forma:

nr. linie FOR variabilă = exp1 TO exp2

În acest caz valoarea pasului (exp3) se consideră 1. Bucla se execută atîta timp cît valoarea variabilei FOR este mai mică sau egală cu exp2.

Execuția instrucțiunii NEXT constă în adunarea valorii pasului (exp3) la valoarea variabilei FOR și testul dacă noua valoare a variabilei nu depășește valoarea limită (exp2). Dacă valoarea limită este depășită execuția buclei ia sfîrșit, următoarea instrucțiune executată fiind cea de după NEXT.

În interiorul unei bucle FOR pot exista alte bucle cu condiția ca buclele să nu se intersecteze.

Exemplu corect

```
10 FOR X=
   50 FOR Y=
      100 FOR Z=
      . . .
      170 NEXT Z
   300 NEXT Y
700 NEXT X
```

Exemplu incorect

```
10 FOR X=
   70 FOR Y=
120 NEXT X
   300 NEXT Y
```

Sînt permise maximum trei bucle FOR incluse una în alta.

Exemplu:

```
100 FOR I=1 TO3
120 FOR J=1 TO 20 STEP 1
130 READ B (I, J)
140 NEXT J
150 NEXT I
```

Instrucțiunea 130 READ se va executa de 20×3=60 de ori.

9.4.10. Exerciţii

1. Programul următor afişează setul complet de caractere.

```
10 FOR N=0 TO 127          30 NEXT N
20 PRINT CHR$ (N)          40 STOP
```

Să se modifice programul, utilizînd funcţia AT în PRINT şi afişînd numai cifre, pentru a se obţine un ceas electronic. (Temporizarea se poate realiza cu un ciclu FOR).

2. Pentru calculul sumei a N numere reale se poate utiliza programul :

```
10 PRINT "N=";             35 S=S+A
15 INPUT N                 40 C=C+1
20 C=1                     50 IF C<=N THEN 30
25 S=0                     60 PRINT S
30 INPUT A
```

Să se descrie acelaşi algoritm de calcul folosind instrucţiunile de ciclare FOR/NEXT cu variabila contor C.

3. Următorul program afişează numărul cu valoarea maximă dintr-un şir de 5 numere introduse :

```
10 INPUT M                 50 M=A
20 FOR I=2 TO 5            60 NEXT I
30 INPUT A                 70 PRINT "MAXIMUL=" ; M
40 IF M>=A THEN 60
```

Programul poate fi utilizat pentru aflarea maximului dintr-un şir de 10 sau 100 de valori dacă modificăm limita 5 din instrucţiunea 20FOR. Ce trebuie modificat în program pentru a calcula minimul.

4. Să se modifice programul de calcul al sumei (din exerciţiul 2) astfel încît să calculeze produsul numerelor nenule din cele N numere (pentru simplificare se va folosi ciclul FOR/NEXT).

5. Pentru calculul factorialului dintr-un număr natural dat poate fi utilizat programul :

```
10 PRINT "INTRODUCETI N=";   40 P=P*I
20 INPUT N                   50 NEXT I
25 P=1                       60 PRINT "N!=" ; P
30 FOR I=2TO N
```

Să se găsească o altă soluţie pentru această problemă.

6. Executaţi programul :

```
10 FOR N=10 TO 1 STEP-1
20 PRINT N
30 NEXT N
```

Transformaţi acest program în unul ce nu conţine ciclul FOR-NEXT (procedînd invers exerciţiul 2).

7. Fie dat un vector A de zece elemente. Programul de mai jos, ordonează crescător elementele acestui vector, deci, în final va fi satisfăcută relaţia $A(1) \leq A(2) \leq \cdots \leq A(10)$.

```
10 FOR I=1 TO 10
15 READ A (I)
20 NEXT I
25 FOR I=1 TO 9
30 FOR J=1 TO 10-I
```

```
40 IF A (J)<=A (J+1) THEN 70
50 T=A (J)
55 A (J)=A (J+1)
60 A (J+1)=T
70 NEXT J
80 NEXT I
85 REM "AFISAREA REZULTATU-
   LUI"
90 FOR I=1 TO 10
100 PRINT A (I).
110 NEXT I
120 END
```

De remarcat că, aceeaşi variabilă I poate apare în mai multe cicluri FOR-NEXT, fără conflict, dacă ciclurile respective nu se intersectează (nu au instrucţiuni comune). De ce a fost necesară utilizarea variabilei T în instrucţiunile 50 şi 60 ?.

Cum poate fi accelerat acest algoritm ? Să se indice un alt algoritm (program) pentru această problemă ? Ciclurile (10—20) pentru citire şi (90—110) pentru tipărire pot fi înlocuite prin cîte o singură instrucţiune matriceală (vezi capitolul : instrucţiuni de calcul cu matrici). Să se modifice programul astfel încît să ordoneze un vector A cu N elemente, $2 \leq N \leq 254$.

8. Similar cu exerciţiul 7, se prezintă un program ce ordonează crescător (sortează alfabetic) o listă de nume, care sînt depuse într-o matrice de şiruri de caractere. Programul se deosebeşte de cel precedent prin faptul că utilizează instrucţiuni IF pentru ciclare, iar ordonarea se face descrescător, însă, afişarea făcîndu-se începînd cu ultimul şir către primul (maxim), rezultatele vor fi afişate în ordine alfabetică (crescătoare).

```
10 REM "SORTARE ALFABETICĂ"
20 DIM W$ (5, 10)
30 LET B=0
40 LET G=5
50 FOR A=1 TO 5
60 INPUT W$ (A)
70 PRINT W$ (A)
80 NEXT A
85 PRINT
90 LET Z=1
100 B=Z+1
110 IF B>G THEN 190
120 IF W$ (B)>W$ (Z) THEN 150
130 Z=Z+1
140 GOTO 100
150 Q$=W$ (Z)
160 W$ (Z)=W$ (B)
170 W$ (B)=Q$
180 GOTO 130
190 PRINT W$ (G)
200 G=G—1
210 IF G>0 THEN 90
220 END
```

Programul foloseşte tabloul W$ (5, 10) pentru păstrarea a 5 şiruri de maximum 10 caractere fiecare. Pentru extinderea la un număr mai mare de şiruri, trebuie modificate instrucţiunile : 20 DIM, 40 LET şi 50 FOR, scriindu-se în locul cifrei 5 un număr dorit (maximum 254). Se observă că, afişarea rezultatelor se face rînd pe rînd, imediat după obţinerea, pe ultima poziţie în W$, a şirului mai mic. Cum poate fi modificat programul pentru a afişa rezultatul, din W$ (5, 10), după ordonarea sa integrală (ca în exerciţiul precedent) ?.

9.4.11. **Utilizarea subrutinelor.** Cînd este necesară efectuarea de mai multe ori într-un program a aceloraşi calcule (instrucţiuni) pentru date eventual diferite, se poate folosi conceptul de subrutină. Utilizarea subrutinelor conduce la o diminuare a dimensiunii programului, deoarece prelucrările se descriu o singură dată (în cadrul subrutinei) şi pot fi executate de cîte ori este nevoie, prin apelarea subrutinei, eventual cu alte date. Folosirea funcţiilor (SIN, COS,

LOG, GET, PUT etc.) în alcătuirea expresiilor, constituie un exemplu în acest sens, deoarece funcţiile sînt definite o singură dată, în interpretor, şi sînt apelate cu diferite argumente de către programator.

Pentru definirea unor subrutine mai puternice (ce descriu prelucrări complexe asupra datelor), în limbajul BASIC sînt disponibile instrucţiunile GOSUB, RETURN şi CALL care vor fi descrise în continuare.

Instrucţiunile GOSUB şi RETURN. Aceste instrucţiuni permit utilizarea subrutinelor în BASIC.

Atunci cînd o succesiune dată de instrucţiuni, apare de mai multe ori într-un program, în locuri diferite, ea poate fi scrisă ca subrutină.

GOSUB
Format:

nr. linie GOSUB n1

Execuţia instrucţiunii constă în transferul controlului la instrucţiunea cu numărul n1.

Instrucţiunea n1 poate fi oricare dintre instrucţiunile ce alcătuiesc o subrutină (în general prima).

RETURN
Format:

nr. linie RETURN

Execuţia instrucţiunii constă în transferul controlului la instrucţiunea ce urmează după instrucţiunea GOSUB, care a apelat subrutina.

Exemplu:

```
100 LET X=5          500 Y=3*X
110 GOSUB 500        510 LET Z=1.2*EXP(Y)
120 X=7              520 LET Y=SQR(Z+2)
130 GOSUB 500        530 IF Y<100 THEN 550
140 X=11             540 RETURN
150 GOSUB 500        550 PRINT X, Y
160 STOP             560 RETURN
```

Variabilele în BASIC sînt globale, deci cele utilizate într-o subrutină pot fi apelate şi în altă parte a programului.

Apelarea subrutinelor scrise în limbaj maşină. Pentru a apela, dintr-un program scris în limbaj BASIC, subrutine scrise în limbaj maşină, se utilizează instrucţiunea CALL.

Formatul instrucţiunii este următorul:

nr. linie CALL (M, P1, P2, . . ., PN)

unde:

— M este numărul subrutinei (de la 0 la 254)

— P1, . . . PN sînt parametrii (constante, variabile sau expresii).

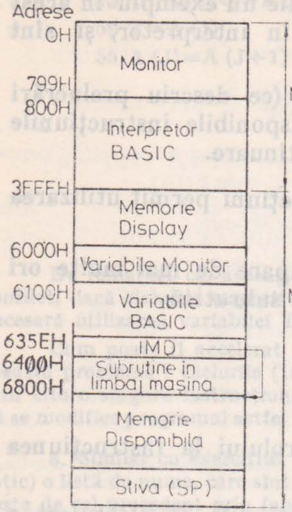


Fig. 9.1. Harta ocupării memoriei.

Legătura dintre numărul subrutinei şi adresa din memorie, unde se află subrutina, se stabileşte prin intermediul unei tabele care se află la o adresă prestabilită (fig. 9.1, harta ocupării memoriei). O intrare în tabelă conţine trei octeţi. În primul octet se află numărul rutinei iar în următorii doi adresa. Sfîrşitul tabelei este detectat prin întîlnirea unei intrări cu numărul 255. În cazul cînd nu se folosesc subrutine în limbaj maşină, apelate dintr-un program BASIC, tabela se reduce la un singur octet cu valoarea 255. În acest caz, iniţializarea tabelei (octetului) se face automat, de către interpretorul BASIC, la lansarea sa în execuţie, prin comenzile B sau GØ8ØØ (vezi paragraful 9.3.1).

Transferul parametrilor se asigură prin referinţă. Adresele parametrilor (pe doi octeţi) sînt depuse în stivă, indicată de registrul SP, în ordinea : adresă de întoarcere în programul BASIC, adresa P1, adresa P2, . . . , adresa PN.

Deci subrutinele în limbaj maşină, în cazul în care folosesc parametrii (primesc şi/sau returnează), vor trebui să facă uz de instrucţiuni POP, pentru obţinerea adreselor parametrilor şi să se termine cu instrucţiunea RET.

Observaţii :

1. Subrutinele în limbaj maşină trebuie să descarce din stivă adresele tuturor parametrilor înainte de terminare (RET), pentru ca adresa luată din stivă, la execuţia instrucţiunii RET, să fie cea depusă de interpretor. Interpretorul nu face nici un control în acest sens.

2. Parametrii au valori reale, reprezentate pe patru octeţi în formatul cu virgulă mobilă. În subrutina scrisă în limbaj maşină, trebuie să se ţină seama de această reprezentare, pentru ca prelucrarea să fie corectă.

3. În cazul cînd parametrii sînt nume de tablou (vector sau matrice) elementele tabloului sînt depuse în memorie liniar, pe coloane (în cazul matricilor), fiecare element ocupînd patru octeţi. La subrutină se transmite adresa primului element de pe prima coloană. Ştiind acest lucru, se poate avea acces în cadrul subrutinei la oricare element al tabloului.

4. Parametrii variabile simple, nedefinite în prealabil (care vor primi valori rezultate din subrutină), nu trebuie să fie intercalaţi cu parametrii expresii.

Exemplu :

În continuare se arată cum trebuie actualizat indicatorul la memoria disponibilă (IMD)* şi definită tabela de legături, pentru utilizarea a trei subrutine cu numerele 1, 2 şi 4.

---

* IMD este iniţializat cu valoarea 6800H, deci indiferent dacă se utilizează subrutine, se rezervă o zonă de 1Ko (400H) pentru subrutine.

```
           ORG     635EH      ; adresa lui IMD *
IMD ;      DW      MDISP      ; actualizare IMD
TSUB :     DB      1          ; intrare pentru
           DW      SUB1       ; subrutina nr. 1
           DB      2
           DW      SUB2       ; adresa subrutinei 2
           DB      4          ; intrare pentru
           DW      SUB3       ; subrutina nr. 4
           DB      255        ; sfîrşit tabelă
SUB1 :     .                  ; cod pentru
           .                  ; subrutina nr. 1
           .
           RET
SUB2 :     .                  ; cod pentru
           .
           .                  ; subrutina nr. 2
           RET
SUB3 :     .                  ; cod pentru
           .
           .                  ; subrutina nr. 4
           RET
MDISP      EQU     $          ; început memorie
           END                ; disponibilă
```

Codul rezultat în urma asamblării programului de mai sus, scris în limbaj de asamblare, va trebui să fie introdus la adresa specificată, cu ajutorul comenzilor MONITOR, înainte de lansarea în execuţie a interpretorului BASIC. După introducerea subrutinelor în limbaj maşină interpretorul BASIC se lansează în execuţie cu comanda GØ815 (pentru a nu se face iniţializarea implicită a lui IMD, ca în cazul absenţei subrutinelor).

9.4.12. Exerciţii

1. Unele aplicaţii necesită accesul la memoria calculatorului pentru scrierea sau citirea valorii unui octet de la o anumită adresă. După cum se ştie unitatea de memorie adresabilă este octetul, iar procesorul poate adresa 65536 octeţi (64 Ko). Pentru realizarea acestor funcţii, unele versuri ale limbajului BASIC conţin instrucţiuni speciale, de exemplu : PEEK A — funcţie a cărei valoare este conţinutul octetului de la adresa A, cu $\emptyset \leq A \leq 65535$; iar POKE A, V este o instrucţiune ce înscrie în octetul de la adresa A valoarea V, cu $\emptyset \leq V \leq 255$. Se vor simula aceste două instrucţiuni utilizînd două subrutine în limbaj de asamblare apelate cu CALL din BASIC. Astfel, pentru PEEK A, se defineşte subrutina numărul 1, cu parametrii A şi B ; A va conţine adresa iar în B se va depune valoarea octetului de la adresa A. Pentru POKE A, V se defineşte a doua subrutină cu parametrii A şi V care va înscrie la adresa A valoarea V ($\emptyset \leq V \leq 255$).

Subrutinele se vor apela, dintr-un program scris în BASIC, prin CALL (1, A, B) pentru PEEK A şi CALL (2, A, V) pentru POKE A, V.

Subrutinele în limbaj de asamblare sînt precedate de tabela de legătură :

```
           ORG     635EH      ; adresa IMD
IMD :      DW      SFSUB      ; actualizare IMD
           DB      1          ; tabela de legătură
           DW      PEEK       ; adresa subrutinei PEEK
           DB      2          ; intrare pentru a
           DW      POKE       ; 2-a subrutină
           DB      255        ; sfîrşitul tabelei
PEEK :     POP     B          ; în reg. B, C adresa celui
                              ; de-al 2-lea parametru
```

* În cazul cînd subrutinele nu depăşesc 1Ko, se poate lăsa valoarea implicită pentru IMD (6800H), începîndu-se cu ORG 64ØØH şi definirea tabelei TSUB

```
        POP     D       ; în D, E adresa variabilei
                        ; A (primul parametru)
        PUSH    B       ; salvare reg. B, C
        CALL    ADRDE   ; în reg. D, E adresa de la care citim
        POP     B       ; în reg, B, C adresa la care depunem
                        ; valoarea citită
; valoarea va trebui memorată în formatul cu virgulă mobilă
        LDAX    D       ; în reg. A octetul citit
        MVI     L, 8    ; normalizare
NORM:   ORA     A
        JZ      NULFL   ; valoarea este zero
        JM      GATA
        RAL
        DCR     L
        JMP     NORM
NULFL:  MVI     L, 40H  ; exponentul lui zero
GATA:   STAX    B       ; scrierea rezultatului în
        INX     B       ; virgulă mobilă
        XRA     A
        STAX    B
        INX     B
        STAX    B
        MOV     A, L    ; exponentul
        INX     B
        STAX    B
        RET
POKE:   ; subrutină care scrie valoarea celui de-al
        ; doilea parametru la adresa dată de primul
        POP     B       ; în B, C adresa valorii
        LDAX    B       ; conversie în întreg
        ORA     A
        JZ      CONT
        MOV     D, A    ; salvare în reg. D
        INX     B
        INX     B
        INX     B
        LDAX    B       ; citire exponent
        ANI     0FH
        MOV     E, A
        MOV     A, D
        RLC
        DCR     E
        JNZ     $—2
; în reg. A se află valoarea 0≤V≤255, convertită în întreg
        POP     D       ; adresa primului parametru
        MOV     B, A    ; salvare valoare
        CALL    ADRDE
        MOV     A, B    ; depunerea valorii la
        STAX    D       ; adresa din reg. D, E
        RET
ADRDE:  ; subrutină ce primeşte în
        ; D, E adresa unui număr în
        ; virgulă mobilă şi întoarce în
        ; D, E valoarea sa (ca număr întreg)
        LDAX    D       ; în reg. A primul octet al
        ORA     A       ; mantisei. Este O?
        JZ      ZERO    ; numărul este zero
```

```
          MOV    H, A
          INX    D
          LDAX   D
          MOV    L, A     ; în (H, L) primii 2 octeţi ai
                          ; numărului
          INX    D
          INX    D
          LDAX   D        ; în reg. A exponentul
          ANI    1FH      ; se păstrează ultimii
          MOV    C, A     ; 5 biţi, salvare în reg. C
          LXI    D, Ø     ; D, E=Ø
CONV:     DAD    H
          XCHG
          ADC    HL, HL   ; HL=HL+HL+CY (Z8Ø)
          DCR    C
          XCHG
          RZ
          JMP    CONV
ZERO:     LXI    D, Ø
          RET
SFSUB     EQU    $
          END
```

Codul obiect rezultat în urma asamblării subrutinelor, se va depune în memoria calculatorului (cu comanda Substitute (S) a monitorului). Apoi, se lansează în execuţie interpretorul BASIC, cu comanda GO8ØØ. În continuare, se poate introduce un program BASIC care utilizează subrutinele în limbaj maşină :

```
1Ø PRINT "INTRODUCETI ADRESA A=";
2Ø INPUT A
25 PRINT "INTRODUCETI VALOAREA V=";
3Ø INPUT V
4Ø REM "ÎN OCTETUL DE LA ADRESA A"
45 REM "VOM DEPUNE VALOAREA V"
5Ø CALL (2, A, V)
60 REM "PENTRU TEST VOM CITI"
65 REM "VALOAREA SCRISA SI O VOM AFIŞA"
7Ø CALL (1, A, C)
8Ø PRINT C
9Ø GOTO 1Ø
```

Programul următor realizează aceeaşi funcţie ca şi comanda Fill a monitorului

```
1Ø INPUT A1, A2, V
2Ø FOR A=A1 TO A2
3Ø CALL (2, A, V)
4Ø NEXT A
5Ø END
```

Pentru ca programele să funcţioneze corect, trebuie ca adresele introduse să corespundă zonei de memorie R/W şi să nu altereze programul BASIC (să nu se autodistrugă).

În cazul utilizării subrutinelor în limbaj maşină, comanda de ştergere a programului : SCR, va şterge şi subrutinele în limbaj maşină. Pentru a nu introduce din nou subrutinele maşină de la tastatură, ele pot fi salvate pe casetă, cu comanda SAVE, imediat după ce au fost introduse şi s-a intrat în BASIC. În

acest caz, ele vor putea fi încărcate de pe casetă cu comanda LOAD înainte de introducerea unui program BASIC, de la tastatură, care le utilizează.

2. Calculatorul dispune de un difuzor care este utilizat ca „martor sonor" la apăsarea unei taste. Dacă se asociază tastelor sunete corespunzătoare notelor muzicale, se va transforma calculatorul într-un instrument muzical.

Difuzorul este conectat prin portul de ieşire 22H (34 zecimal) pe bitul 3. Pentru a obţine frecvenţa unei note dorite, se va transmite pe acest port şiruri de 1 şi Ø de durate corespunzătoare. Pentru a se transmite o valoare pe un port de ieşire poate fi utilizată funcţia PUT (nr. port) din BASIC, de exemplu :

```
1Ø PUT (34)=Ø
2Ø PUT (34)=8 (1 în bitul 3)
3Ø GOTO 1Ø
```

Pentru a obţine frecvenţe mai mari, trebuie să se utilizeze o subrutină în limbaj de asamblare. Astfel, se va defini o subrutină în limbaj de asamblare care va primi ca parametrii : frecvenţa notei dorite şi durata şi va genera un sunet corespunzător. Subrutina se va apela cu instrucţiunea CALL (1, F, D), după ce, în F s-a încărcat frecvenţa iar în variabila D durata. Valorile lui F şi D depind de modul în care s-a scris subrutina de generare şi de frecvenţa impulsurilor de ceas ale calculatorului.

Se va folosi următoarea subrutină în limbaj maşină :

```
         ORG    635EH
         DW     MDISP   ; adresa memoriei libere
         DB     1       ; Tabela de legătură
         DW     SUNET   ; adresa subrutinei
         DB     255     ; sfîrşitul tabelei
SUNET :  POP    H       ; adresa celui de al doilea parametru (P2)
         CALL   CVINT   ; conversie în întreg
         MOV    E, A    ; valoarea P2 (durata) o păstrăm în reg. A
         POP    H       ; adresa P1 (frecvenţa)
         CALL   CVINT   ; conversie în întreg
         MOV    B, A    ; salvare P1 în reg. B
LOOP1    MOV    D, B    ; în D frecvenţa
LOOP2 :  MVI    A, 8    ; transmite 1 pe bitul 3
         OUT    22H     ; al portului 22H
         DCR    D
         JNZ    LOOP2
         MOV    D, B    ; în D frecvenţa
LOOP3 :  XRA    A       ; transmite Ø pe bitul 3
         OUT    22H     ; al portului 22H
         DCR    D
         JNZ    LOOP3
         DCR    E       ; decrementează durata
         JNZ    LOOP1
         RET
CVINT :  ; subrutină ce primeşte în reg. H, L adresa unui număr reprezentat
         ; în formatul virgulă mobilă pe 4 octeţi şi întoarce
         ; valoarea sa (întreagă) în reg. A (se presupune că valoarea
         ; numărului este cuprinsă între Ø şi 255).
         MOV    A, M    ; primul octet al numărului
         ORA    A       ; este Ø ?
         RZ
         MOV    D, A    ; salvarea primului octet
         INX    H       ; în reg. D
```

```
        INX   H
        INX   H
        MOV   A, M    ; exponentul
        ANI   OFH     ; păstrează ultimii 4 biţi
        MOV   C, A
        MOV   A, D    ; conversie în întreg
        RLC           ; (aliniere la dreapta)
        DCR   C
        JNZ   $—2
        RET
MDISP:  EQU   $
        END
```

După introducerea codului subrutinei cu comanda Substitute a Monitorului, se va lansa în execuţie interpretorul BASIC cu comanda GO8ØØ. Subrutina se va apela, aşa cum s-a arătat, cu instrucţiunea CALL din BASIC (de exemplu CALL (1, F, D)).

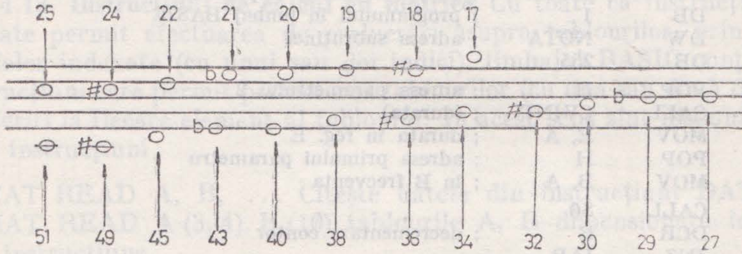


Fig. 9.2. Parametrii de frecvenţă ai notelor.

Pentru a obţine (genera) notele dorite, parametrul de frecvenţă F, va avea valorile din figura 9.2. Programul următor asociază tastelor numerice 1, 2, 3, 4 notele Do, Re, Mi, Fa.

```
5 D=5Ø
7 F=51
1Ø A$=INKEY$
15 IF A$=" "THEN 25
2Ø ON VAL (A$) GOSUB 3Ø, 4Ø, 5Ø, 6Ø
25 CALL (1, F, D)
27 GOTO 1Ø
3Ø F=51
35 RETURN
4Ø F=45
45 RETURN
5Ø F=4Ø
55 RETURN
6Ø F=38
7Ø RETURN
```

O notă va dura pînă la acţionarea unei taste (1—4), corespunzătoare altei note.

Încercaţi să extindeţi programul, încît să asociaţi şi celelalte note unor taste.

Programul următor generează note folosind generatorul de numere aleatoare pentru frecvenţă şi durată :

```
1Ø CALL (1, INT (RND (Ø)*9Ø+1Ø), INT (RND (Ø)*5Ø+5))
2Ø GO TO 1Ø
```

Generarea de note cu frecvenţe din ce în ce mai mari se poate realiza cu programul :

```
1Ø FOR I=1ØØ TO 1Ø STEP-1
2Ø CALL (1, I, 5Ø)
3Ø NEXT I
```

Observaţie: Ştergerea programelor de mai sus, în vederea introducerii altora ce utilizează subrutina în limbaj de asamblare, nu se poate face cu SCR pentru că se va şterge şi subrutina. Pentru a nu şterge programele linie cu linie, se va ieşi de sub controlul interpretorului BASIC acţionînd RESET, intrînd sub controlul Monitorului, iar apoi se va reintra în BASIC cu comanda G∅815. Astfel, subrutina în limbaj maşină va rămîne şi se va şterge doar programul BASIC.

3. Se va utiliza o subrutină în limbaj maşină ce permite generarea de sunete de durată mai mare, folosindu-se pentru durată un contor pe doi octeţi. Subrutina va primi tot doi parametrii, primul reprezentînd frecvenţa iar cel de-al doilea durata.

```
        ORG   635EH
        DW    MDISP    ; adresa memoriei disponibile
        DB    1        ; programului în limbaj BASIC
        DW    NOTA     ; adresa subrutinei
        DB    255
NOTA:   POP   H        ; adresa parametrului 2
        CALL  CVINT    ; (durata)
        MOV   E, A     ; durata în reg. E
        POP   H        ; adresa primului parametru
        MOV   B, A     ; în B frecvenţa
IAR:    CALL  S∅
        DCR   E        ; decrementare contor
        JNZ   IAR
        RET
S∅:     LXI   H,35FFH
S1:     MOV   D, B     ; în D frecvenţa
        MVI   A, 8     ; 1 în bitul 3
        OUT   22H
S2:     DCX   H        ; buclă în care se
        MOV   A, H     ; generează 1 pe durata
        ANA   A        ; dată de reg. D.
        RZ
        DCR   D
        JNZ   S2
        MOV   D, B
        MVI   A, 0     ; ∅ în bitul 3
        OUT   22H
S3:     DCX   H        ; buclă în care
        MOV   A, H     ; se generează ∅
        ANA   A        ; pe durata dată de
        RZ
        DCR   D        reg. D
        JNZ   S3
        JMP   S1
```

Subrutina CVINT este identică cu cea din exemplul precedent.

Programul următor va apela subrutina în limbaj de asamblare descrisă mai sus pentru interpretarea unei melodii. Frecvenţele şi duratele sînt date în instrucţiuni DATA, fiind citite cu instrucţiunea READ.

```
1∅ READ A, B
2∅ CALL (1, A, B)
```

```
30 GO TO 10
100 DATA 51 3, 45, 1, 40, 3, 51, 1, 40, 2
110 DATA 51, 2, 40, 4, 45, 3, 40, 1, 38, 2, 40, 1
120 DATA 45, 1, 38, 8, 40, 3, 38, 1, 34, 3
130 DATA 40, 1, 34, 2, 40, 2, 34, 4, 38, 3
140 DATA 34, 1, 30, 2, 34, 1, 38, 30, 8, 34, 3
150 DATA 51, 1, 45, 1, 40, 1, 38, 1, 34, 1
160 DATA 30, 8, 30, 3, 45, 1, 40, 1, 38, 1
170 DATA 34, 1, 30, 1, 27, 8, 27, 3, 40, 1
180 DATA 38, 1, 34, 1, 30, 1, 27, 1, 25, 7
190 DATA 27, 1, 30, 2, 38, 2, 27, 2, 34, 2, 25, 6
200 END
```

Execuţia programului se va termina prin apariţia erorii 21, deoarece au fost epuizate toate constantele din instrucţiunile DATA. Modificaţi programul utilizînd instrucţiunea RESTORE, reluînd interpretarea melodiei.

9.4.13. Instrucţiuni de calcul cu matrice. Cu toate că instrucţiunile deja prezentate permit efectuarea de prelucrări asupra tablourilor, prin utilizarea variabilelor indexate (cu unul sau doi indici), limbajul BASIC conţine un set de instrucţiuni care permit prelucrarea tablourilor (cu una sau două dimensiuni) fără referiri la fiecare element al tabloului. În acest sens sînt disponibile următoarele instrucţiuni :

MAT READ A, B, ... MAT READ A (3,4), B (10)	Citeşte datele din instrucţiuni DATA pentru tablourile A, B dimensionate în prealabil sau în instrucţiune.

Observaţie : Datorită prefixului MAT, utilizat în instrucţiuni, tablourile vor fi numite matrice (indiferent dacă au una sau două dimensiuni).

MAT INPUT A, B, ··· MAT INPUT A (2, 4), B (3, 3), ···	Citeşte datele pentru matricile A, B de la consolă. Datele introduse sînt memorate linie cu linie, în matrice.
MAT PRINT A, B, ··· MAT PRINT A (4, 4), B (3, 2), ···	Tipăreşte linie cu linie valorile curente ale matricilor A, B dimensionate în prealabil sau redimensionate în instrucţiune.
MAT A=B	Matricea A primeşte dimensiunile lui B şi apoi B este copiat în A.
MAT A=B+C MAT A=B—C	Adună sau scade matricile B şi C. Matricile B şi C trebuie să aibă aceleaşi dimensiuni. Matricea A are dimensiunile matricii rezultat.
MAT A=B*C	Înmulţeşte matricile B şi C şi depune rezultatul în matricea A, care va primi dimensiunile rezultatului. Dimensiunile matricilor B şi C trebuie să fie compatibile (produs de matrici).
MAT A=(expresie)*B	Realizează produsul matricei B cu un scalar egal cu valoarea expresiei dintre paranteze. Rezultatul produsului este depus în matricea A, care este dimensionată corespunzător.
MAT A=INV (B) MAT A=INV (B), D	Inversează matricea pătrată B. Inversa este memorată în matricea A care va avea dimensiunea lui B. Valoarea determinantului va fi atribuită variabilei D.
MAT A=TRN (B)	Calculează transpusa matricei B şi depune valoarea în matricea A, care va avea dimensiunile matricei rezultat. Tablourile A şi B trebuie să fie distincte.

MAT A=ZER MAT A=ZER (3, 4) MAT A=ZER (7)	Atribuie tuturor elementelor matricei A valoarea zero. Matricea A poate fi redimensionată conform specificaţiilor din instrucţiune.
MAT A=CON MAT A=CON (3, 10) MAT A=CON (8)	Atribuie tuturor elementelor matricei A valoarea 1. Matricea A poate fi redimensionată conform specificaţiilor din instrucţiune.
MAT A=IDN MAT A=IDN (2, 5) MAT A=IDN (7)	Atribuie elementelor diagonalei principale valoarea 1, celorlalte elemente li se atribuie valoarea zero. Dacă A este matrice pătrată ea va deveni matrice unitate.

Observaţie :

Instrucţiunile matriciale pot fi împărţite în două categorii :

— instrucţiuni de intrare/ieşire

MAT READ
MAT INPUT
MAT PRINT

— instrucţiuni de atribuire — în care matricei din membrul stîng i se atribuie rezultatul unei operaţii matriciale.

Matricile introduse prin instrucţiunile de intrare/ieşire sau matricea din membrul stîng al unei instrucţiuni de atribuire, pot fi alocate la execuţia instrucţiunii MAT respective, nefiind necesară declararea lor în instrucţiunea DIM. În cazul în care matricile au fost declarate în prealabil, într-o instrucţiune DIM, sau au fost introduse printr-o instrucţiune MAT anterioară, ele pot fi redimensionate cu respectarea condiţiilor :

— numărul de dimensiuni ale tabloului să fie păstrat (un vector nu poate deveni matrice şi nici invers),

— numărul de elemente al matricei redimensionate trebuie să nu depăşească numărul de elemente al matricei iniţiale.

Dimensiunile specificate în instrucţiunile matriciale pot fi constante, variabile sau expresii.

Exemplu :

MAT READ A (M+N, N+1), B

Citirea matricelor. Citirea matricelor se realizează cu instrucţiunile : MAT READ şi MAT INPUT.

```
20 MAT READ B (2, 3), A (3)
.
.
.
30 DATA 5, 11, —17, 1, 2, 3, 1E7, 0, 1
```

După execuţia instrucţiunii 20 matricile A şi B vor conţine :

B (1, 1)=5 ; B (1, 2)=11 ; B (1, 3)=—17
B (2, 1)=1 ; B (2, 2)=2 ; B (2, 3)=3
A (1)=10^7 ; A (2)=0 ; A (3)=1.

Matricile A şi B puteau fi declarate şi în instrucţiunea DIM caz în care, în instrucţiunea 20, nu mai trebuiau specificate dimensiunile

```
10 DIM A (3), B (2, 3)
20 MAT READ B, A
.
.
```

Aceleaşi observaţii sînt valabile şi pentru instrucţiunea MAT INPUT. În acest caz, datele sînt citite de la consolă (tastatură)

```
10 DIM A (10), B (3, 3)
20 MAT INPUT A (3), B (2, 3)
.
.
```

Matricile A, B vor fi alocate la execuţia instrucţiunii DIM. Instrucţiunea MAT INPUT va redimensiona matricile A şi B şi apoi va citi de la consolă valorile elementelor (pe linii).

Se observă că redimensionarea satisface cele două cerinţe :

— păstrarea tipului (vector sau matrice),

— noul număr de elemente este mai mic decît cel alocat iniţial.

Observaţie : Instrucţiunea 10 DIM nu era absolut necesară, matricile puteau fi alocate în instrucţiunea MAT INPUT.

Tipărirea matricelor. Pentru tipărirea matricelor se utilizează instrucţiunea MAT PRINT. Matricile se tipăresc linie cu linie. În cadrul unei linii, spaţierea între elemente se face conform separatorului utilizat în lista de matrici din MAT PRINT (separatori pot fi „," sau „;"). După tipărirea unei linii a matricei se lasă o linie liberă, pentru ca liniile să apară mai clar. Dacă lista din MAT PRINT conţine mai multe matrici, între liniile unei matrici şi ale următoarei din listă, se lasă două linii libere.

```
10 DIM A (3, 2)
20 MAT READ A, B (3)
30 MAT PRINT A, B
40 DATA 1, 2, 3, 4, 5, 6, 7, 8, 9
50 END
```

Instrucţiunea MAT READ va citi tablourile A şi B.

La execuţia instrucţiunii 30 se vor tipări următoarele valori la consolă (display) :

```
1          2
3          4                    ; Matricea A (3, 2)
5          6
7          ; Vectorul B (3) va fi tipărit ca
8          ; matrice coloană B (3, 1)
9
```

Elementele matricei A pot fi tipărite comasat, dacă se foloseşte separatorul „;". În acest caz, instrucţiunea 30 va fi :

```
30 MAT PRINT A ; B
```

B va fi tipărit la fel ca mai sus.

În cazul cînd în MAT PRINT, se specifică dimensiuni pentru un tablou din listă, tabloul va fi mai întîi redimensionat şi apoi tipărit cu noile dimensiuni. De exemplu, dacă instrucţiunea 30 de mai sus ar fi fost :

```
30 MAT PRINT A (2, 2), B (2)
```

atunci s-ar fi tipărit :

1	5	; matricea A (2, 2)
3	2	
7		; vectorul B(2)
8		

Observație : La redimensionarea unei matrice, trebuie avut în vedere faptul că elementele matricei sînt memorate liniar, formînd un vector din coloane puse cap la cap. Deci, elementele de pe aceeași poziție, în matrice nu vor fi mereu aceleași după redimensionare. De exemplu, matricea :

$$A(4,3)=\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \\ 10 & 11 & 12 \end{bmatrix}$$

redimensionată ca A (3, 3) va fi :

$$A(3,3)=\begin{bmatrix} 1 & 1\emptyset & 8 \\ 4 & 2 & 11 \\ 7 & 5 & 3 \end{bmatrix}$$

Calculul inversei unei matrice. O matrice pătrată, nesingulară poate fi inversată. De remarcat că, o matrice poate fi inversată în ea însăși, adică, matricea din membrul stîng al instrucțiunii poate fi aceeași cu cea din membrul drept.

Exemplu : 1Ø MAT A=INV (A) este corectă.

Dacă se dorește și calculul determinantului matricei A (sau numai determinantul) se va folosi instrucțiunea : 1Ø MAT A=INV (A), V1.

La execuția instrucțiunii 1Ø, variabilei V1 i se va atribui valoarea determinantului matricei A.

```
1Ø MAT READ A (2, 2)
2Ø MAT B=INV (A), D
3Ø MAT C=A*B
4Ø MAT PRINT A, B, C
5Ø PRINT D
6Ø DATA 1, 2, 3, 4
7Ø END
```

Execuția acestui program va determina tipărirea matricelor :

$$A=\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}, \quad B=A^{-1}=\begin{bmatrix} -2 & 1 \\ 1.5 & -\emptyset.5 \end{bmatrix}$$

$$C=A*B=A*A^{-1}=\begin{bmatrix} 1 & \emptyset \\ \emptyset & 1 \end{bmatrix}$$ — matrice unitate

Se va tipări și valoarea determinantului matricei A

D=det (A)=(1*4)−(2*3)=−2

Observaţie : Pentru calculul inversei unei matrice, se scrie transpusa matricei, se înlocuieşte fiecare element cu complementul sau algebric (minorul cu semnul corespunzător) împărţit la valoarea determinantului matricei. Inversa se poate calcula mai eficient prin alte metode.

Transpusa unei matrice. Transpusa unei matrice A (M, N) va avea dimensiunile (N, M). Un vector fiind tratat ca matrice coloană, transpusa sa va fi o matrice linie.

```
10 MAT INPUT V (10)
20 MAT A=TRN (V)
30 END
```

Instrucţiunea 10 va citi de la consolă vectorul V, de 10 elemente. Instrucţiunea 20 va aloca matricea A (1, 10) şi-i va atribui valorile elementelor vectorului V. Transpusa unei matrice linie este o matrice coloană.

```
10 DIM V (10)
20 MAT INPUT A (1, 10)
30 MAT V=TRN (A)
40 END
```

În acest program transpusa matricei linie A va fi memorată în vectorul V.

Observaţii :

1. În instrucţiunile matriceale se poate lucra cu vectori atîta timp cît operaţiile efectuate nu necesită transformarea vectorilor (tablouri cu o dimensiune) în matrice (tablouri cu două dimensiuni). Acest lucru (permiterea utilizării vectorilor în instrucţiunile matriceale) este util în descrierea majorităţii problemelor în care se prelucrează matrici (ex. : sisteme de ecuaţii liniare — termenii liberi şi necunoscutele sînt notaţi ca vectori B (N), X (N) nu ca matrice B (N, 1), X (N, 1))

2. O matrice nu poate fi transpusă în ea însăşi. Instrucţiunea :

```
10 MAT A=TRN (A) este incorectă.
```

Produsul a două matrice. Pentru a putea înmulţi două matrice numărul de coloane al primei matrice trebuie să fie egal cu numărul de linii al celei de-a doua.

Se consideră instrucţiunea :

```
50 MAT A=B*C
```

Dacă B are dimensiunile (P, N), iar C (N, Q), matricea A va avea dimensiunile (P, Q).

În cazul în care matricea A nu a fost în prealabil alocată, ea va fi alocată la execuţia instrucţiunii 50, cu dimensiunile (P, Q). Dacă matricea A a fost alocată ea va căpăta dimensiunile (P, Q).

Observaţie : Matricea rezultat nu poate figura ca matrice factor. Deci instrucţiunile :

```
10 MAT B=B*C
sau
10 MAT C=B*C
```

sînt incorecte.

Adunarea şi scăderea matricelor. Pentru a putea aduna sau scădea două matrice, ele trebuie să aibă aceleaşi dimensiuni. Aceleaşi dimensiuni vor fi atribuite şi matricei rezultat.

Într-o instrucţiune nu se poate executa decît o singură operaţie. Instrucţiunea :

10 MAT A=B+C—D este greşită.

Matricea din membrul stîng al atribuirii, poate figura şi în membrul drept.

10 MAT A=A+B este corectă

Înmulţirea unei matrice cu un scalar.

10 MAT A=(expresie)*B

Matricea A va primi dimensiunile lui B şi va avea ca elemente, elementele matricei B, înmulţite cu valoarea expresiei. Expresia trebuie să fie inclusă între paranteze

Exemplu : 30 MAT A=(COS (x)+SIN (x))*B

Iniţializarea unei matrice. Generarea unei matrice cu toate elementele zero se poate realiza cu instrucţiunea :

50 MAT A=ZER (3,2)

Dacă matricea A a fost alocată înainte de execuţia instrucţiunii 50 va primi dimensiunile (3, 2), dacă nu, va fi alocată cu aceleaşi dimensiuni.

În cazul în care se urmăreşte iniţializarea matricei, fără alterarea dimensiunilor, nu se vor mai specifica dimensiunile, în instrucţiunea MAT

```
10 DIM A (5, 5)
.
.
50 MAT A=ZER
```

Pentru tablouri cu o singură dimensiune se poate utiliza :

10 MAT V=ZER (10)

Pentru a genera o matrice cu toate elementele 1, se pot folosi, de exemplu, instrucţiunile :

```
60 MAT B=CON (3, 3)
sau
70 MAT B=CON
```

Instrucţiunea 70 va fi utilizată în cazul în care B este alocat în prealabil.

Pentru a genera o matrice unitate (cu unu pe diagonala principală şi zero în rest) se va folosi funcţia IDN. Exemplu :

10 MAT A=IDN (3,3)

Dacă matricea nu este pătrată (sau nu se specifică dimensiuni egale în funcţia IDN), prin diagonală principală se va înţelege, diagonala ce pleacă din

elementul cu poziţia (1, 1) şi merge diagonal pînă la epuizarea numărului de linii sau de coloane.

De exemplu :

```
10 MAT A=IDN(2,3)
```

va genera :

$$A=\begin{bmatrix} 1 & \emptyset & \emptyset \\ \emptyset & 1 & \emptyset \end{bmatrix}$$

9.4.14. **Instrucţiuni de prelucrare grafică.** Soluţiile multor probleme constau din şiruri lungi de numere a căror interpretare este destul de dificilă. Reprezentarea sub formă grafică a acestor şiruri de valori numerice facilitează aprecierea cantitativă şi calitativă a soluţiilor.

Problemele în care se utilizează facilităţile grafice impun reprezentarea grafică a unor valori numerice (tablouri, valori ale unor funcţii etc.) sau realizarea unor desene, hărţi etc.

În primul caz utilizatorul este interesat de forma graficului şi de încadrarea lui pe ecranul dispozitivului de afişare. În cel de-al doilea caz, pentru utilizator va fi importantă specificarea explicită a scării de reprezentare, pentru a efectua, eventual, măsurători pe desene etc.

Una din soluţiile de realizare a graficelor folosind limbajul BASIC poate consta în scrierea unor subrutine în limbaj de asamblare, care utilizează un anumit display, şi utilizarea lor, în BASIC, cu ajutorul instrucţiunii CALL. Această soluţie este puţin flexibilă (nu este independentă de tipul perifericului grafic) şi destul de greoaie pentru un începător.

De aceea au fost introduse instrucţiuni speciale pentru prelucrări grafice.

Instrucţiunea MOVE. Formatul instrucţiunii :

nr. linie MOVE X, Y

unde : X, Y pot fi constante, variabile sau expresii.

Instrucţiunea MOVE este folosită pentru a poziţiona spotul în punctul de coordonate (X, Y). De menţionat că instrucţiunea MOVE execută numai poziţionarea în punctul de coordonate (X, Y) nu marchează punctul respectiv.

Instrucţiunea DRAW. Formatul instrucţiunii este :

nr. linie DRAW X, Y

unde : X, Y pot fi constante, variabile sau expresii.

Instrucţiunea DRAW este folosită pentru a trage o linie între punctul în care se află spotul la întîlnirea instrucţiunii şi punctul de coordonate X, Y specificate în instrucţiune.

De exemplu dacă se dorește reprezentarea datelor conținute în vectorul V (N) se poate folosi secvența

```
.
.
.
100 MOVE 1, V (1)
110 FOR J=1TO N
120 DRAW J, V (J)
130 NEXT J
.
.
```

Instrucțiunea 100 execută o poziționare în punctul de coordonate (1, V (1)), apoi în ciclul FOR se unesc prin linii elementele vectorului V. Se observă că instrucțiunile MOVE și DRAW sînt suficiente pentru a face reprezentări grafice.

Instrucțiunile WINDOW și VIEWPORT. Coordonatele X, Y specificate în instrucțiunile MOVE și DRAW sînt exprimate în unitățile de măsură în care sînt exprimate mărimile de reprezentat (metri, kg, volți etc.).

Acest sistem de coordonate definește „spațiul utilizatorului" sau spațiul virtual. Spațiul virtual este limitat practic de precizia aritmeticii mașinii ($\pm 10_{18}$). Acest „spațiu utilizator" va fi reprezentat la o anumită scară pe suprafața display-ului.

După cum s-a arătat la începutul paragrafului, în problemele de reprezentare grafică a unor date, utilizatorul preferă să definească limitele spațiului în care sînt cuprinse valorile de prezentat și să accepte scara și originea implicit determinate, fără a mai specifica o scară de reprezentare și o origine.

Instrucțiunea prin care utilizatorul specifică limitele „spațiului utilizator" în care sînt cuprinse datele de reprezentat este :

nr. linie WINDOW A, B, C, D

unde : A, B, C, D sînt variabile, constante sau expresii care reprezintă limitele spațiului utilizator, în ordinea :

A = limita stîngă	C = limita inferioară
B = limita dreaptă	D = limita superioară

Cele patru limite definesc un spațiu dreptunghiular. Orice punct de coordonate (X, Y) pentru care :

$A \leqslant X \leqslant B$

$C \leqslant Y \leqslant D$

va fi reprezentat grafic. Punctele care cad în afara dreptunghiului nu vor avea imagine pe suprafața display-ului. Din linia trasă cu instrucțiunile :

```
10 MOVE X1, Y1
20 DRAW X2, Y2
```

va fi reprezentată numai porţiunea interioară dreptunghiului (ferestrei), definite mai sus, de limitele A, B, C, D.
Observaţie : De remarcat că poate fi luat implicit spaţiul utilizator maxim posibil (impus de aritmetică)

```
10 WINDOW —10↑18, 10↑18, —10↑18, 10↑18
```

În acest caz toate valorile numerice cu care se poate lucra într-un program sînt reprezentate grafic. Dezavantajul constă în aceea că scara implicată este extrem de mare, şi un domeniu de valori destul de mare ($\approx 10^6$) se va reprezenta printr-un singur punct (pe un display cu latura de 1000 puncte).

Laturile ferestrei (dreptunghiului) declarate în instrucţiunea WINDOW trebuie să fie cît mai apropiate de domeniul de valori de reprezentat, pentru ca graficul obţinut să fie cît mai fin.

Pînă acum s-a considerat că spaţiul utilizator, definit prin instrucţiunea WINDOW, va fi reprezentat pe toată suprafaţa display-ului. În unele aplicaţii se va dori scrierea unor comentarii alături de grafic sau realizarea mai multor grafice pe aceeaşi suprafaţă.

Pentru a descrie porţiunea din suprafaţă display-ului pe care va fi realizat graficul (va fi proiectat spaţiul utilizator) se foloseşte instrucţiunea :

```
nr. linie VIEWPORT A, B, C, D
```

unde : A, B, C, D sînt variabile, constante sau expresii şi reprezintă limitele zonei din suprafaţa display-ului, în aceeaşi ordine ca pentru WINDOW : stîngă, dreaptă, inferioară, superioară.

Spre deosebire de WINDOW unităţile în care se exprimă limitele A, B, C, D sînt unităţi fizice.

Alegerea unităţii fizice în care să se exprime limitele A, B, C, D trebuie să satisfacă cerinţele de independenţă faţă de tipul perifericului grafic, (unele display-uri au suprafaţa pătrată, altele dreptunghiulare etc.). Independenţa de tipul perifericului, asigură ca un program scris pentru un anumit display să se poată folosi şi pentru alte tipuri de echipamente de afişare.

Unitatea în care se exprimă limitele A, B, C, D s-a ales ca fiind un procent din latura pătratului cel mai mare, care poate fi înscris în suprafaţa display-ului. Aceasta se va numi unitate grafică (UG). Originea suprafeţei display-ului se consideră în colţul din stînga jos. În aceste condiţii instrucţiunea :

```
10 VIEWPORT 0, 100, 0, 100
```

va specifica dimensiunile celui mai mare pătrat înscris în suprafaţa display-ului (dacă suprafaţa display-ului este pătrată, va reprezenta întreaga suprafaţă).

Această instrucţiune este executată implicit de sistem.

De asemenea implicit este executată şi instrucţiunea :

```
WINDOW 0, 100, 0, 100
```

Aceste două instrucţiuni realizează o corespondenţă unu la unu între unităţile utilizator (pe orizontală şi pe verticală) şi unitatea grafică (UG).

Cu aceste inițializări implicite un cerc va apărea nedistorsionat, deoarece pe ambele axe de coordonate se folosesc aceleași unități de măsură, iar suprafața graficului este pătrată (aceeași scară pe ambele axe).

Programul următor va trasa un cerc pe orice display :

```
100 MOVE 85, 50                              130 NEXT J
110 FOR J=0TO 2*PI STEP PI/10                140 END
120 DRAW 35*COS (J)+50, 35*SIN (J)+50
```

Cercul va avea centrul în punctul de coordonate (50, 50), iar raza va fi 35.

Se observă că, în program, nu a fost necesară utilizarea instrucțiunilor WINDOW sau VIEWPORT. Nu este necesară cunoașterea particularităților display-ului. Este suficient să se știe că orice display are o suprafață de cel puțin 100/100 unități (UG). Evident, pentru reprezentări de date care nu variază între 0 și 100 se impune utilizarea instrucțiunii WINDOW (altfel se reprezintă doar porțiunea din grafic cuprinsă în fereastra 0, 100, 0, 100). Dacă se va dori ca reprezentarea graficului să se facă numai pe o porțiune din suprafața display-ului, va trebui utilizată instrucțiunea VIEWPORT. De pildă, pentru colțul din dreapta sus se va folosi instrucțiunea :

```
10 VIEWPORT 75, 100, 70, 100
```

Instrucțiunile prezentate mai sus : MOVE, DRAW, WINDOW, VIEWPORT formează setul minim de instrucțiuni grafice, cu ajutorul cărora se poate rezolva orice problemă de reprezentare grafică. Sînt prevăzute, de asemenea, instrucțiuni care înlesnesc utilizarea subrutinelor pentru realizarea graficelor, manipularea figurilor (rotații, translații) și instrucțiuni care realizează independența programelor față de tipul perifericului grafic.

Instrucțiunile RMOVE și RDRAW. Formatul instrucțiunilor :

```
nr. linie RMOVE X, Y
nr. linie RDRAW X, Y
```

unde : X, Y pot fi constante, variabile sau expresii.

Instrucțiunile RMOVE și RDRAW se deosebesc de MOVE și DRAW prin faptul că coordonatele X, Y nu sînt raportate la originea sistemului, ci la poziția spotului (punctului grafic), la momentul execuției instrucțiunii. Aceste instrucțiuni sînt utile în scrierea subrutinelor care generează diverse figuri. Figurile respective vor putea fi desenate în orice loc, pe suprafața display-ului, prin utilizarea instrucțiunii MOVE, înainte de apelul rutinei care generează figura. Astfel, se realizează translația figurilor.

Instrucțiunea SCALE. În unele aplicații, realizarea de hărți, diagrame, desene etc., este utilă indicarea explicită a scării de reprezentare dorite.

Formatul instrucțiunii folosite în acest caz este :

```
nr. linie SCALE S1, S2
```

unde : S1, S2 sînt constante, variabile sau expresii care reprezintă factorii de scară pe orizontală şi pe verticală. Factorul de scară indică numărul unităţilor utilizator reprezentate pe o unitate grafică

$$S_i = \frac{\text{unităţi utilizator}}{\text{unităţi grafice (UG)}}.$$

Observaţie : În cazul în care se doreşte efectuarea de măsurători pe grafic, utilizatorul va trebui să ştie cîţi milimetri (de exemplu) reprezintă o unitate grafică (UG) pentru perifericul respectiv (se reaminteşte că 1UG reprezintă a suta parte din latura pătratului maxim inscriptibil în suprafaţa display-ului).

Originea sistemului de coordonate se consideră în punctul în care se află spotul (punctul grafic) la execuţia instrucţiunii SCALE. Pentru aceasta se va folosi, eventual, instrucţiunea MOVE înaintea instrucţiunii SCALE. Punctul specificat ca origine va fi reprezentat pe grafic în colţul din stînga jos.

Se poate remarca faptul că instrucţiunea WINDOW este echivalentă cu SCALE, avînd în plus specificarea originei. Coordonatele originii vor lua locul parametrilor A, C din instrucţiunea WINDOW, iar factorii de scară vor determina limitele B şi D din WINDOW.

Instrucţiunea ROTATE. Formatul instrucţiunii ROTATE este următorul :

nr. linie ROTATE U

unde : U este o constantă, variabilă sau expresie.

Instrucţiunea ROTATE are efect numai asupra instrucţiunilor RMOVE şi RDRAW, realizînd rotaţia cu unghiul U a vectorilor generaţi cu RDRAW sau a poziţionării realizate cu RMOVE.

Dacă instrucţiunea ROTATE este utilizată înainte de apelul unei rutine care generează o figură cu ajutorul instrucţiunilor RMOVE şi RDRAW, figura generată va apărea rotită cu unghiul specificat (în radiani). Rotaţia figurii respective va fi realizată faţă de punctul în care s-a început generarea figurii, punctul unde se află spotul la începutul secvenţei de program (rutinei) ce desenează figura. Instrucţiunile MOVE şi DRAW nu sînt afectate de ROTATE, deoarece coordonatele absolute, specificate în aceste instrucţiuni trebuie să rămînă nealterate, pentru a se putea executa poziţionări în punctele dorite înainte de generarea unor figuri cu RMOVE şi RDRAW. Astfel, se permite utilizarea simultană şi independentă a translaţiei şi rotaţiei figurilor.

Exemplu : Următorul program va trasa un pătrat cu latura de 10 unităţi (Se consideră iniţializările implicite deci nu se vor folosi WINDOW şi VIEWPORT)

```
90 MOVE 0, 0            120 RDRAW —10, 0
100 RDRAW 10, 0         130 RDRAW 0, —10
110 RDRAW 0, 10         140 END
```

Pătratul va fi trasat începînd din origine. Instrucţiunea 90 poziţionează spotul în originea spaţiului utilizator care pentru iniţializările implicite, coincide cu originea suprafeţei grafice (colţul din stînga jos).

Pătratul poate fi micşorat, mărit sau transformat în dreptunghi cu ajutorul instrucţiunii SCALE S1, S2. Dacă S1=S2≠1 pătratul va fi micşorat sau mărit, dar laturile vor rămîne egale între ele. Pentru S1=S2=1 pătratul va fi

trasat neschimbat față de cel din exemplul de mai sus. În cazul în care $S1 \neq S2$ laturile trasate pe display nu vor mai fi egale, iar pătratul va fi reprezentat ca dreptunghi.

Următorul program va desena pătratul de două ori mai mare decît cel trasat de programul precedent.

```
1Ø MOVE Ø, Ø
2Ø SCALE 1/2, 1/2
3Ø GOSUB 1ØØ
4Ø STOP
1ØØ RDRAW 1Ø, Ø
11Ø RDRAW Ø, 1Ø
12Ø RDRAW —1Ø, Ø
13Ø RDRAW Ø, —1Ø
14Ø RETURN
15Ø END
```

Se observă că secvența de program care desena pătratul a fost scrisă ca subrutină. Dacă se dorește ca pătratul să fie desenat în alt loc pe suprafața display-ului se va schimba instrucțiunea 10. De exemplu, punînd :

```
1Ø MOVE 5Ø, 5Ø
```

pătratul va fi trasat (în sens trigonometric) începînd din punctul de coordonate (5Ø, 5Ø).

Observație : Latura pătratului va avea tot 1Ø unități utilizator (s-a folosit aceeași secvență de instrucțiuni pentru generarea pătratului), însă datorită indicării scării de 1/2, dimensiunea laturii pătratului desenat va fi 2Ø unități grafice (UG), deci dublă față de latura pătratului trasat de programul precedent.

Pătratul generat de subrutina care începe cu instrucțiunea 1ØØ poate fi trasat începînd din punctul de coordonate (5Ø, 5Ø) și rotit cu unghiul PI/4 cu ajutorul programului :

```
1Ø MOVE 5Ø, 5Ø
2Ø ROTATE PI/4
3Ø GOSUB 1ØØ
4Ø STOP
```

În cazul în care se dorește ca pătratul să fie micșorat și rotit, în sens invers trigonometric, se va putea folosi programul :

```
1Ø MOVE 2Ø, 3Ø
2Ø SCALE 2, 2
3Ø ROTATE —PI/3
4Ø GOSUB 1ØØ
5Ø STOP
```

Pătratul va fi trasat începînd din punctul de coordonate (2Ø, 3Ø), micșorat la jumătate (latura de 5UG) și rotit în sens orar cu unghiul PI/3.

La instrucțiunea 1ØØ începe aceeași subrutină de generare a pătratului ca și în programele precedente.

Instrucțiunea INIT. Formatul instrucțiunii este următorul :

```
nr. linie INIT [P]
```

Execuția instrucțiunii INIT constă în ștergerea ecranului display-ului și efectuarea inițializărilor grafice implicite, anume :

```
WINDOW Ø, 1ØØ, Ø, 1ØØ
```

și

```
VIEWPORT Ø, 1ØØ, Ø, 1ØØ
```

Deci instrucţiunile grafice SCALE, WINDOW, VIEWPORT şi ROTATE executate înainte de INIT îşi pierd valabilitatea.

Dacă parametrul P este prezent, atunci display-ul trece în mod pagină, altfel rămîne în mod defilare.

Instrucţiunea GSINPUT. Formatul instrucţiunii GSINPUT este următorul:

nr. linie GSINPUT V1, V2

unde : V1, V2 sînt variabile.

Instrucţiunea GSINPUT este utilizată pentru a realiza independenţa totală faţă de tipul perifericului grafic. Execuţia instrucţiunii constă în atribuirea variabilelor V1, V2 a dimensiunilor exprimate în unităţi grafice (UG) ale suprafeţei display-ului. De exemplu, un display poate avea suprafaţa dreptunghiulară cu latura orizontală mai mare sau cu cea verticală mai mare (fig. 9.3 a, b).

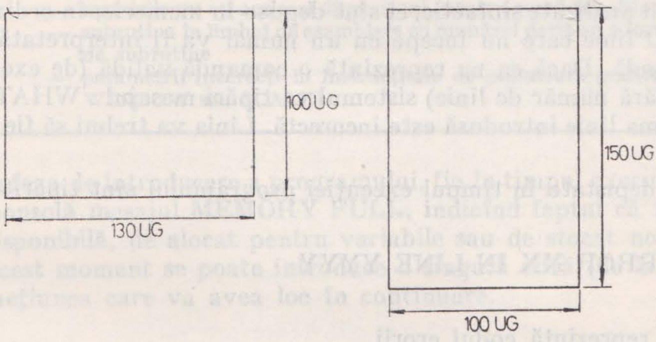


Fig. 9.3. Stabilirea unităţilor grafice pe suprafaţa ecranului de afişare : a) V1=13Ø UG ; V2=1ØØ UG ; b) V1= =1ØØ UG ; V2=15ØUG.

Executînd instrucţiunea GSINPUT V1, V2 pentru display-ul din figura 9.3. a, variabilele V1, V2 vor primi valorile V1=13Ø, V2=1ØØ, iar pentru display-ul din figura 9.3 b, V1=1ØØ, V2=15Ø.

Utilizînd variabilele V1, V2 în instrucţiunea VIEWPORT se vor obţine programe care utilizează întreaga suprafaţă a display-ului, indiferent de forma ei :

```
100 GSINPUT A, B
110 VIEWPORT Ø, A, Ø, B
```

Instrucţiunile PLOT şi UNPLOT. Instrucţiunile PLOT şi UNPLOT împreună cu caracterele speciale (semigrafice), utilizabile în instrucţiunea PRINT, oferă facilităţi semigrafice limbajului BASIC. Formatul instrucţiunilor este :

nr. linie PLOT X, Y

şi

nr. linie UNPLOT X, Y

uned : X, Y pot fi constante, variabile sau expresii.

Execuția instrucțiunilor constă în a aprinde sau stinge pătrățelul de coordonate X, Y de pe suprafața display-ului. Suprafața display-ului este formată din 64×64 pătrățele, un pătrățel conținînd 4×4=16 puncte grafice. Originea suprafeței, (pătrățelul de coordonate Ø, Ø) se consideră în colțul din stînga, jos al suprafeței display-ului. Prin urmare, valorile expresiilor X și Y vor fi cuprinse între Ø și 63. Următorul program, trasează două linii, una orizontală, cealaltă verticală, care se intersectează în mijlocul ecranului

```
1Ø INIT                    4Ø PLOT I, 32
2Ø FOR I=Ø TO 63           5Ø NEXT I
3Ø PLOT 32, I              6Ø END
```

9.5. Mesajele de eroare ale interpretorului BASIC

În timpul introducerii programului, instrucțiunile (liniile ce încep cu un număr) nu sînt analizate sintactic, ci sînt depuse în memorie, în ordinea numerelor de linie. O linie care nu începe cu un număr va fi interpretată de sistem drept o comandă. Dacă ea nu reprezintă o comandă validă (de exemplu este o instrucțiune fără număr de linie) sistemul va tipări mesajul : WHAT ?, indicînd faptul că ultima linie introdusă este incorectă. Linia va trebui să fie reintrodusă corect.

Erorile depistate în timpul execuției programului sînt tipărite la consolă sub forma :

ERROR XX IN LINE YYYY

unde :

— XX reprezintă codul erorii

— YYYY reprezintă numărul liniei în care a fost depistată eroarea

Semnificațiile codurilor de eroare (XX) posibile, sînt date în continuare :

Codul erorii	Semnificație
1	— programul nu se termină cu instrucțiunile END sau STOP
2	— tip de instrucțiunea nerecunoscut (cuvînt cheie incorect)
3	— există instrucțiuni sursă după instrucțiunea END
4	— numărul liniei destinație este incorect (în instrucțiunile IF, GOTO, GOSUB, ON)
5	— numărul liniei destinație este inexistent (în instrucțiunile IF, GOTO, GOSUB, ON)
6	— caracter ilegal
7	— instrucțiune neterminată
8	— expresie incorectă
9	— eroare la conversia în virgulă mobilă
1Ø	— utilizare incorectă a funcțiilor GET sau PUT
11	— tentativă de redimensionare (prin DIM) a unui tablou
12	— tablou utilizat înainte de a fi definit (stocat)
13	— argumentul funcțiilor SIN, COS sau TAN prea mare (>1Ø⁶)
14	— relație incorectă în instrucțiunea IF
15	— depășirea stivelor în evaluator — expresie prea complicată

16	— eroare în ridicarea la putere (0 la puterea 0 sau număr negativ la putere reală)
17	— instrucţiune FOR utilizată fără instrucţiunea NEXT corespunzătoare (buclă FOR neînchisă)
18	— instrucţiune NEXT utilizată fără instrucţiunea FOR corespunzătoare
19	— depăşire stivă FOR (peste 3 cicluri cuprinse unul într-altul)
20	— indice mai mare de 254 sau egal cu 0
21	— se doreşte citirea mai multor constante decît au fost definite în instrucţiunile DATA
22	— depăşirea valorilor declarate ale indicilor
23	— radical din număr negativ
24	— logaritm din număr negativ
25	— dimensiuni necompatibile petru produs de matrici
26	— matricea din membrul stîng al egalităţii nu poate apărea în membrul drept (pentru transpusă sau produs de matrici)
27	— matrice singulară — nu poate fi inversată
28	— redimensionare incorectă — noile dimensiuni depăşesc pe cele maxime
29	— prin redimensionare nu poate fi schimbat numărul de dimensiuni al unui tablou
30	— matricea nu este patrată — nu poate fi inversată
31	— matricile nu au aceleaşi dimensiuni (pentru sumă sau diferenţă)
32	— subrutina în limbaj de asamblare cu numărul dorit nu a fost găsită în tabela de subrutine
33	— parametrii incorecţi în instrucţiune de prelucrare grafică (VIEWPORT, WINDOW sau SCALE)

Fie în faza de introducere a programului, fie în timpul execuţiei sale poate apărea la consolă mesajul MEMORY FULL, indicînd faptul că nu mai există memorie disponibilă, de alocat pentru variabile sau de stocat noi linii de program. În acest moment se poate introduce o singură cifră (de la consolă) care va indica acţiunea care va avea loc în continuare.

Număr introdus	Semnificaţie	
0	RUN	— execută programul din memorie
1	SAVE	— copiază programul din memorie pe casetă
2	LIST	— listează programul din memorie
3	SCR	— şterge programul din memorie

Capitolul 10. Microcalculatorul aMIC în matematicile elementare și statistică

Dar, mai întîi, o vedere de ansamblu a programării „personale" pe aMIC.

Utilizarea echipamentelor de calcul într-o serie de activități economico-sociale a fost mult timp frînată de regimul de lucru specific puterii de calcul centralizate : prelucrarea lucrărilor pe loturi. Timpul de răspuns, adică durata între momentul predării spre execuție a unei lucrări și momentul primirii rezultatului prelucrării, poate fi de ordinul orelor sau chiar zilelor, funcție de o serie de factori specifici exploatării pe loturi la centrele de calcul. Valoarea mare a timpului de răspuns implică două inconveniente majore :

a) apariția de discontinuități mari în activitatea de elaborare, testare și depanare a programelor, avînd ca rezultat o eficiență scăzută a activității de programare.

b) imposibilitatea utilizării puterii de calcul în activități care solicită o colaborare permanentă om-calculator pe durata execuției programelor.

Regimul de lucru cu acces direct al utilizatorilor la echipamentul de calcul, fie că este vorba de acces local sau de la distanță, rezolvă în cea mai mare parte inconvenientele de mai sus, însă această rezolvare nu este în beneficiul unui număr mare de utilizatori. Principalul obstacol în răspîndirea acestui mod de lucru îl constituie costul ridicat al unui sistem de calcul interactiv și numărul limitat de terminale ce pot fi gestionate simultan în cadrul unui asemenea sistem.

Apariția microcalculatoarelor, considerate ca fiind sisteme total interactive, permite diseminarea puterii de calcul pentru orice tip de aplicații, prin avantajele specifice pe care le posedă : preț de cost scăzut, programare în limbaje conversaționale de nivel înalt, posibilitatea conectării prin linii de teletransmisie la sisteme de calcul de capacitate medie-mare, interfață simplă cu operatorul uman etc.

Microcalculatorul personal aMIC pune la dispoziția utilizatorilor săi o serie de facilități destinate implementării cît mai simple a puterii de calcul într-un domeniu vast de aplicații.

Interfața cu operatorul este concepută pe două niveluri : nivelul monitor și nivelul BASIC. Ambele au făcut obiectul unei tratări detaliate în capitolele precedente. Reliefăm doar faptul că această ierarhizare, neobișnuită la microcalculatoarele personale, permite dezvoltarea de aplicații în două direcții : cea a domeniilor specializate, necesitînd elaborarea de programe în limbaj de asamblare, și cea a domeniilor de larg interes, scrierea aplicațiilor făcîndu-se în limbajul BASIC-aMIC, al căror obiect îl constituie prezentul capitol.

Limbajul BASIC implementat pe microcalculatorul aMIC este simplu şi uşor de învăţat, fiind accesibil unei categorii largi de utilizatori, chiar nespecialişti în informatică. Crearea, modificarea şi testarea programelor se desfăşoară rapid, datorită modului de lucru conversaţional al interpretorului BASIC ; numărul de erori posibile semnalate la execuţia programelor este relativ redus, depanarea programelor neridicînd astfel probleme deosebite. Corectarea liniilor de program eronate se poate executa imediat, programul fiind disponibil pentru o nouă rulare, ciclul modificare-testare fiind foarte scurt. Acest mod de lucru, gen „încearcă să vezi ce se întîmplă" nu este cel mai indicat, dar prezintă nete avantaje pentru nespecialişti, eliminînd o bună parte din bariera psihologică ridicată de utilizarea unui instrument nou de lucru în cadrul activităţii clasice a acestora.

Domeniile de utilizare includ cu preponderenţă calcule matematice complexe, prelucrării de tip gestiune economică cu un volum mic-mediu de date, aplicaţii tehnico-ştiinţifice, învăţămînt, aplicaţii grafice etc. Toate exemplele date în continuare au mai mult un caracter demonstrativ, didactic, încercîndu-se acoperirea unui număr cît mai mare de domenii diferite.

Pe măsură ce programarea în BASIC este însuşită, utilizatorul îşi poate defini propriile sale aplicaţii plecînd de la exemplele demonstrative ; partea cea mai dificilă o constituie abordarea algoritmică a aplicaţiei ce se doreşte a fi transpusă pe aMIC. Scopul acestui capitol este şi acela de a demonstra că un număr surprinzător de activităţi pot fi abordate prin prisma transpunerii lor pe microcalculator, şi că aceasta este practic la îndemîna tuturor.

10.1. Rezolvarea ecuaţiei de gradul II :

$AX^2+BX+C=0$, unde A, B, C sînt numere reale. RE (X), IM (X) sînt partea reală şi partea imaginară ale rădăcinilor complexe conjugate.

Programul se bazează pe organigrama dată în figura 10.1

```
10 PRINT"AX*X+BX+C=O"
15 PRINT"INTRODUCETI COEFICIENTII:"
25 INPUT A,B,C
30 IF A<>O THEN 45
35 PRINT"ECUATIA NU ESTE DE GR.2"
40 GO TO 95
45 M=B*B-4*A*C
50 IFM>=O THEN 80
55 PRINT"ECUATIA ARE RADACINI COMPLEXE"
60 PRINT"PARTEA REALA",B/A/2
70 PRINT"PARTEA IMAGINARA",-SQR(-M)/A/2
72 GO TO 95
80 PRINT"RADACINI REALE"
85 Z=SQR(M)
86 W=2*A
90 PRINT(-B+Z)/W,(-B-Z)/W
95 END
```

```
REZOLVAREA ECUATIEI
AX*X+BX+C=O
CU A,B,C NUMERE REALE
INTRODUCETI COEFICIENTII:
3            4            45.6
ECUATIA ARE RADACINI COMPLEXE
PARTEA REALA     .666667
PARTEA IMAGINARA              -3.8413
```

```
REZOLVAREA ECUATIEI
AX*X+BX+C=O
CU A,B,C NUMERE REALE
INTRODUCETI COEFICIENTII:
0            23           1.1
ECUATIA NU ESTE DE GR.2
```

```
REZOLVAREA ECUATIEI
AX*X+BX+C=O
CU A,B,C NUMERE REALE
INTRODUCETI COEFICIENTII:
12           13.12        -45
RADACINI REALE
1.46551      -2.55884
```

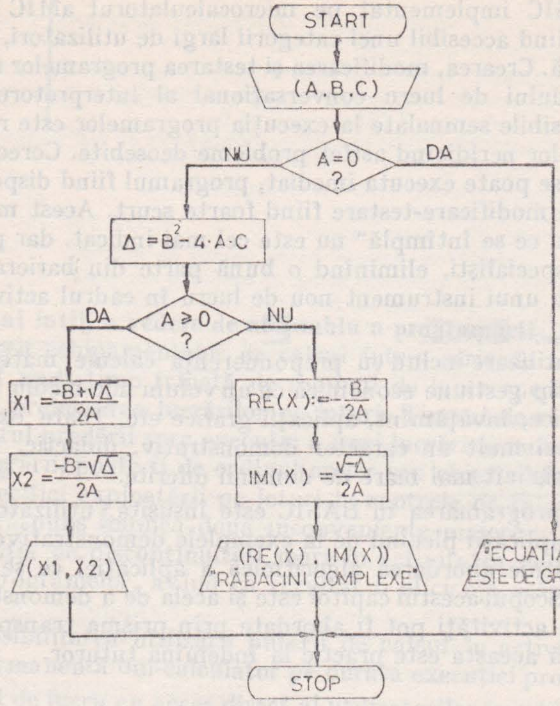


Fig. 10.1. Organigrama rezolvării ecuaţiei de gradul II.

```
5   PRINT"SA SE REZOLVE"
10  PRINT"AX*X+BX+C < O"
15  PRINT"INTRODUCETI COEFICIENTII:"
20  INPUT A,B,C
25  IF A=O THEN 55
30  M=B*B-4*A*C
35 IF M>=O THEN 70
40  PRINT"ECUATIA NU ARE"
45  PRINT"SOLUTIE IN R"
50  GO TO 170
55  PRINT"INECUATIA NU ESTE"
60  PRINT"DE GRADUL 2"
65  GO TO 170
70  V=SQR(M)
75  U=2*A
80  T=(-B+V)/U
85  V=(-B-V)/U
90  IF T < V THEN 110
95  U=V
100 V=T
105 T=U
110 GOSUB 140
115 GO TO 170
140 PRINT"X< ",T," SAU"
150 GO TO 170
160 PRINT T,"< X <",V
165 RETURN
170 END
```

10.2. Rezolvarea inecuaţiei

$AX^2 + BX + C < \emptyset$.

Programul se bazează pe organigrama din figura 10.2.

```
SA SE REZOLVE
AX*X+BX+C < Ø
INTRODUCETI COEFICIENTII:
 2.2            4.1            78.9
ECUATIA ARE
SOLUTIE IN R
```

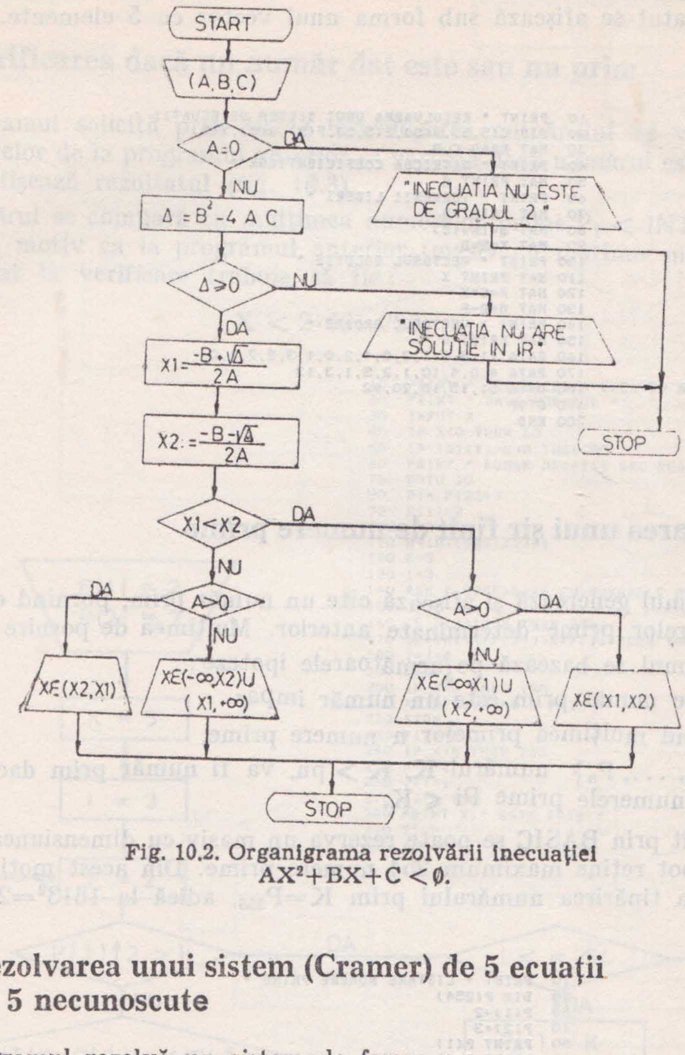


Fig. 10.2. Organigrama rezolvării inecuației
$AX^2+BX+C<\emptyset$.

10.3. Rezolvarea unui sistem (Cramer) de 5 ecuații cu 5 necunoscute

Programul rezolvă un sistem de forma :

$$C \cdot X = D$$

unde :

C este matrice de 5×5 elemente

X este vector coloană de 5 elemente

D este vector coloană de 5 elemente.

Elementele matricilor C și D nu sînt solicitate prin dialog, ele sînt prefixate în program la liniile 16∅, 17∅ și 18∅. Dacă se doresc alte valori, se vor modifica aceste linii.

Rezultatul se afișează sub forma unui vector cu 5 elemente.

```
10  PRINT " REZOLVAREA UNUI SISTEM DE ECUATII "
20  DIM C(5,5),D(5),E(5,5),F(5),G(5),X(5)
30  MAT READ C,D
40  PRINT " MATRICEA COEFICIENTILOR "
50  MAT PRINT C
60  PRINT " TERMENII LIBERI "
70  MAT PRINT D
80  MAT E=INV(C)
90  MAT X=E*D
100 PRINT " VECTORUL SOLUTIE "
110 MAT PRINT X
120 MAT F=C*X
130 MAT G=D-F
140 PRINT " VECTORUL EROARE "
150 MAT PRINT G
160 DATA 11,3,0,1,2,0,4,2,0,1,3,2,7,1,0
170 DATA 4,0,4,10,1,2,5,1,3,13
180 DATA 51,15,15,20,92
190 STOP
200 END
```

10.4. Afișarea unui șir finit de numere prime

Programul generează și afișează cîte un număr prim, pornind de la mulțimea numerelor prime determinate anterior. Mulțimea de pornire este 2,3.

Programul se bazează pe următoarele ipoteze :

1) orice număr prim este un număr impar

2) avînd mulțimea primelor n numere prime,

$\{P_1, P_2, \ldots, P_n\}$ numărul K, K $>$pn, va fi număr prim dacă nu este divizibil cu numerele prime Pi $\leqslant$ K.

Întrucît prin BASIC se poate rezerva un masiv cu dimensiunea maximă de 254, se pot reține maximum 254 numere prime. Din acest motiv afișarea se oprește la tipărirea numărului prim $K=P_{255}$, adică la $1613^2=2\cdot 6\emptyset 1\cdot 769$.

```
10  PRINT " LISTARE NUMERE PRIME "
20  DIM P(254)
30  P(1)=2
40  P(2)=3
50  PRINT P(1)
60  PRINT P(2)
70  K=5
80  I=3
90  J=2
100 IF P(J)^2 > K THEN 180
110 IF INT(K/P(J))=K/P(J) THEN 160
120 J=J+1
130 IF J<=254 THEN 100
140 STOP
150 REM " K NU ESTE PRIM "
160 K=K+2
170 GOTO 90
180 REM " K ESTE PRIM "
190 IF I>254 THEN 210
200 P(I)=K
210 PRINT K
220 K=K+2
230 I=I+1
240 GOTO 90
250 END
```

10.5. Verificarea dacă un număr dat este sau nu prim

Programul solicită prin dialog introducerea numărului de verificat. În baza ipotezelor de la programul anterior se verifică dacă numărul este prim sau nu şi se afişează rezultatul (fig. 1Ø.3).

Numărul se compară cu mulţimea numerelor prime $P_i < INT(\sqrt{X}) + 1$. Din acelaşi motiv ca la programul anterior (masiv cu mărime maximă 254) numărul dat la verificare trebuie să fie:

$$X < 2 \cdot 6Ø1 \cdot 769$$

```
10  PRINT " VERIFICEREA DACA UN NUMAR E PRIM "
20  PRINT " DATI NUMARUL "
30  INPUT X
40  IF X<0 THEN 60
50  IF INT(X)-X=0 THEN 80
60  PRINT " NUMAR NEGATIV SAU REAL "
70  GOTO 30
80  DIM P(254)
90  P(1)=2
100 P(2)=3
110 M=INT(SQR(X))+1
120 K=5
130 I=3
140 REM " COMPARARE CU NUMERELE PRIME "
150 J=2
160 IF P(J)^2>K THEN 230
170 IF INT(K/P(J))=K/P(J) THEN 190
180 J=J+1
190 K=K+2
200 IF K<X THEN 150
210 PRINT X;" NU E PRIM "
220 STOP
230 P(I)=K
240 IF X=K THEN 280
250 K=K+2
260 I=I+1
270 GOTO 150
280 PRINT X;" ESTE PRIM "
290 STOP
300 END
```

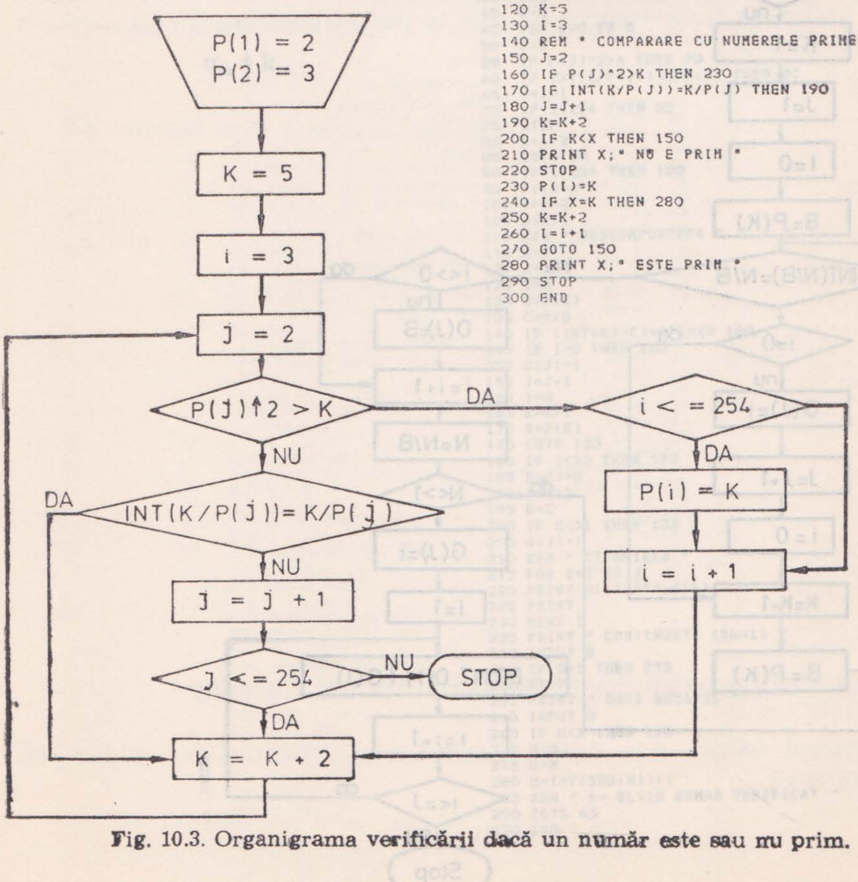


Fig. 10.3. Organigrama verificării dacă un număr este sau nu prim.

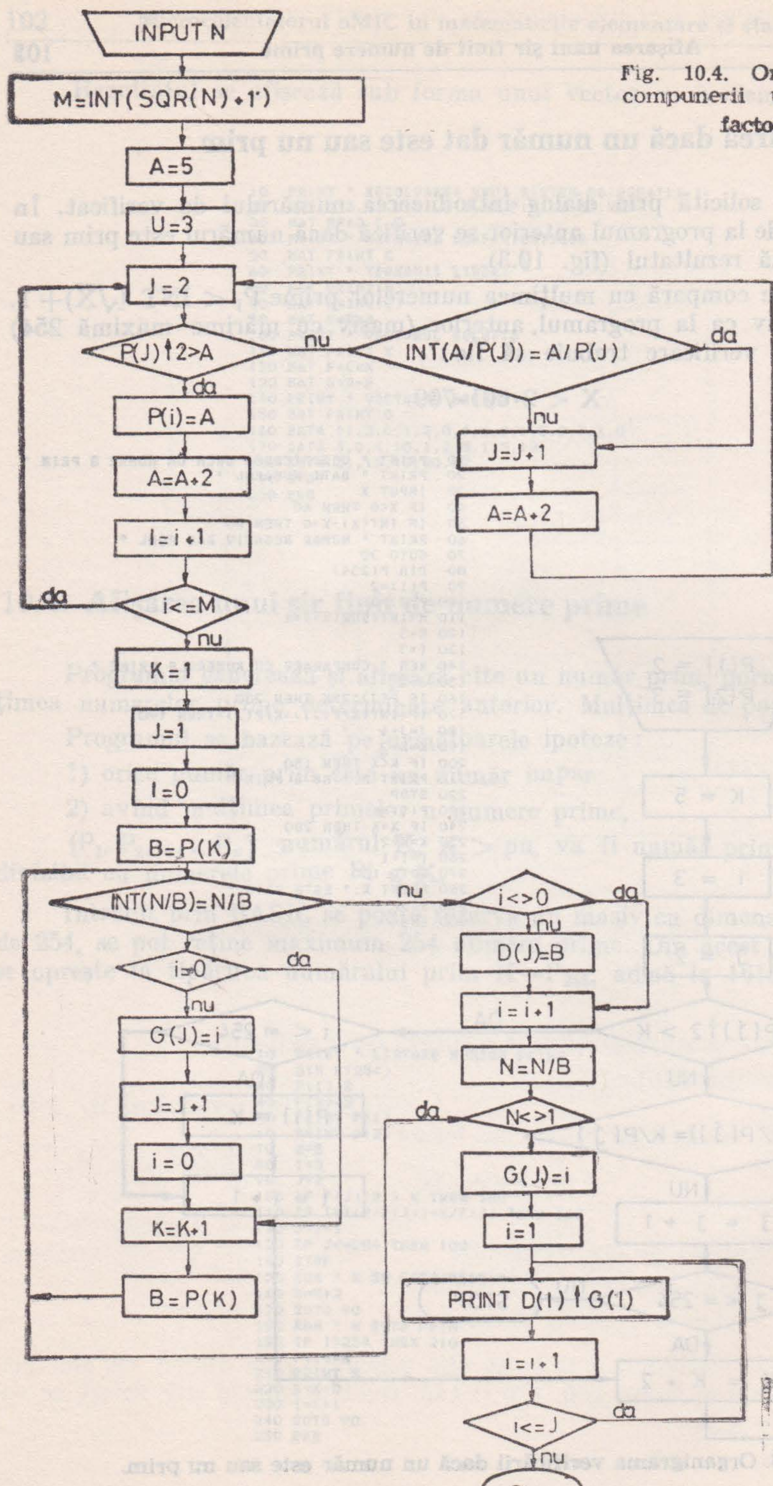


Fig. 10.4. **Organigrama descompunerii unui număr în factori primi.**

10.6. Descompunerea unui număr în factori primi

Programul solicită prin dialog numărul de descompus în factori primi. Se aplică ipotezele de la 10.4 și 10.5 pen ru a obține șirul de numere prime pînă la $\sqrt{N}$, N fiind numărul dat. Se contorizează apoi de cîte ori se cuprinde fiecare număr prim în numărul dat.

Considerînd descompunerea de forma :

$$N = q_1^{k1} \cdot q_2^{k2} \cdot \ldots \cdot q_m^{km}$$

se afișează :

$q_1 \uparrow k_1$

$q_2 \uparrow k_2$

.

.

.

$q_m \uparrow k_m$

```
5   PRINT " DESCOMPUNEREA IN FACTORI PRIMI "
10  DIM P(254),D(20),G(20)
15  PRINT " DATI NUMARUL "
20  INPUT N
25  M=INT(SQR(N))+1
30  P(1)=2
35  P(2)=3
37  U=3
40  A=5
45  FOR I=U TO M
50  J=2
55  IF P(J)^2>A THEN 90
60  IF INT(A/P(J))=A/P(J) THEN 80
65  J=J+1
70  IF J<254 THEN 55
75  STOP
80  A=A+2
85  GOTO 50
90  IF I>254 THEN 100
95  P(I)=A
100 A=A+2
105 NEXTI
110 REM " DESCOMPUNEREA "
115 K=1
120 J=1
125 I=0
130 B=P(K)
135 C=N/B
140 IF (INT(C)-C)=0 THEN 180
145 IF I=0 THEN 165
150 G(J)=I
155 J=J+1
160 I=0
165 K=K+1
170 B=P(K)
175 GOTO 135
180 IF I<>0 THEN 190
185 D(J)=B
190 I=I+1
195 N=C
200 IF C<>1 THEN 135
205 G(J)=I
210 REM " TIPARIREA "
215 FOR I=1 TO J
220 PRINT D(I);"^";G(I)
225 PRINT
230 NEXT I
235 PRINT " CONTINUATI (DA=1) "
240 INPUT D
245 IF D=1 THEN 255
250 STOP
255 PRINT " DATI NUMARUL "
260 INPUT U
265 IF U<N THEN 115
270 N=U
275 U=M
280 M=INT(SQR(N))+1
285 REM " A= ULTIM NUMAR VERIFICAT "
290 GOTO 45
300 END
```

10.7. Determinarea celui mai mare divizor comun

Programul solicită prin dialog cele două numere întregi și afișează cel mai mare divizor comun al lor.

```
5   PRINT " CEL MAI MARE DIVIZOR COMUN "
10  PRINT " DATI CELE DOUA NUMERE "
15  INPUT A,B
20  IF A=B THEN 50
25  IF A>B THEN 40
30  B=B-A
35  GOTO 20
40  A=A-B
45  GOTO 20
50  PRINT A
55  GOTO 10
60  END
```

10.8. Simplificarea unei fracții

Programul solicită prin dialog numărătorul și monitorul fracției. Se afișează un vector cu toate valorile cu care se poate simplifica fracția, inclusiv fracția simplificată.

```
5   PRINT " SIMPLIFICAREA FRACTIILOR "
10  PRINT " DATI NUMARATORUL "
15  INPUT A
20  PRINT " DATI NUMITORUL "
25  INPUT B
30  DIM S(100)
35  I=0
40  D=2
45  IF D<=B THEN 95
50  IF I=0 THEN 80
55  PRINT " SE POATE SIMPLIFICA CU:
60  MAT PRINT S(I)
65  PRINT " FRACTIA SIMPLIFICATA: ";
70  PRINT A;"/";B
75  STOP
80  PRINT " NU SE POATE SIMPLIFICA "
85  PRINT " FUNCTIA DATA: ";
90  GOTO 70
95  C=B/D
100 IF INT(C)=C THEN 115
105 D=D+1
110 GOTO 45
115 E=A/D
120 IF INT(E)=E THEN 130
125 GOTO 105
130 A=E
135 B=C
140 I=I+1
145 S(I)=D
150 GOTO 40
155 END
```

10.9. Calculul aproximativ al factorialului unui număr

Programul solicită prin dialog numărul n și determină factorialul său pe baza formulei:

$$n! \approx \sqrt{2\pi n}\,(n/e)^n$$

Formula este indicată pentru valori mari ale lui n, deoarece nu există înmulţiri succesive, mari consumatoare de timp.

```
5   PRINT " FACTORIAL "
10  PRINT " DATI NUMARUL "
15  INPUT N
20  IF N>O THEN 35
25  PRINT " NUMAR NEGATIV SAU ZERO "
30  STOP
35  F=(SQR(2*PI*N))*(N/EE)^N
40  PRINT N;"! = ";F
45  STOP
50  END
```

10.10. Permutări, aranjamente, combinări

Programul solicită mai întîi tipul operaţiei, şi corespunzător acesteia, parametrii necesari. Calculele se execută după formulele:

a) permutări de n elemente:

$$P_n = n!$$

b) combinări de n elemente luate cîte r:

$$C_n^r = \frac{n!}{(n-r)!\,r!}$$

c) aranjamente de n elemente luate cîte r:

$$A_n^r = \frac{n!}{(n-r)!}$$

Se afişează apoi rezultatul operaţiei.

```
10  PRINT " P= PERMUTARE A N ELEMENTE
20  PRINT " A= ARANJAMENTE A R ELEMENTE DIN N
30  PRINT " C= COMBINARE A N ELEMENTE LUATE CITE R
40  A$=INKEY$
50  IFA$=" " THEN 40
60  IFA$="P" THEN 100.
70  IFA$="A" THEN 100
80  IFA$="C" THEN 100
90  GOTO 10
100 PRINT " INTRODUCETI PE N
110 INPUT N
115 IFA$="P" THEN 170
120 PRINT " INTRODUCETI PE R
130 INPUT R
140 IF R<=N THEN 170
150 PRINT " R > N
160 GOTO 120
170 Z = N
180 GOSUB 400
190 IFA$="P" THEN 340
200 B = A
210 Z = N-R
220 GOSUB 400
230 C = A
240 IFA$="A" THEN 280
250 Z = R
260 GOSUB 400
270 C = C*A
280 PRINT "N";A$;"R = ";B/C
290 PRINT " CONTINUATI (D/N) "
300 A$=INKEY$
310 IFA$=" " THEN 300
320 IFA$="D" THEN 10
330 STOP
340 PRINT "NP = ";A
350 GOTO 290
400 A = 1
410 FOR X=1 TO Z
420 A = A*X
430 NEXT X
440 RETURN
450 END
```

Subprogramul de la linia 400 calculează exact valoarea factorului Z !. Se apelează cu valorile Z egale cu n, n—r, r de care avem nevoie în formulele anterioare.

10.11. Ordonarea unui şir de numere

Programul solicită mai întîi cîte numere se ordonează, apoi şirul de numere efective. Se execută ordonarea în ordine crescătoare sau descrescătoare, afişînd şirul ordonat.

```
5   PRINT " ORDONAREA UNUI SIR DE NUMERE
10  PRINT " IN ORDINE CRESCATOARE/DESCRESCATOARE "
15  PRINT " CITE NUMERE INTRODUCETI ? "
20  INPUT N
25  DIM A(N)
30  PRINT " DATI SIRUL DE NUMERE "
35  MAT INPUT A
40  PRINT " IN ORDINE CRESCATOARE (DA=1)
45  INPUT D
55  FOR I=1 TO N-1
60  FOR J=I+1 TO N
65  IF D=1 THEN 80
70  IF A(I)>A(J) THEN 110
75  GOTO 85
80  IF A(I)<A(J) THEN110
85  B=A(J)
90  FOR K=J TO I+1 STEP -1
95  A(K)=A(K-1)
100 NEXT K
105 A(I)=B
110 NEXT J
115 NEXT I
120 MAT PRINT A
125 STOP
130 END
```

10.12. Calculul sumei celor mai mari numere dintr-un şir de numere date

```
10  PRINT " ADUNAREA A N NUMERE DIN M
20  PRINT " AVIND VALOAREA CEA MAI MARE
30  PRINT " CITE NUMERE INTRODUCETI ? "
40  INPUT M
50  DIM A(M)
60  PRINT " DATI NUMERELE
70  MAT INPUT A
80  PRINT " CITE NUMERE SE ADUNA
90  INPUT N
100 IF N<=M THEN 130
110 PRINT " N > M   SE ADUNA TOATE NUMERELE
120 N=M
130 DIM B(N)
140 FOR I=1 TO N
150 B(I)=A(I)
160 FOR J=I+1 TO M
170 IF B(I)>=A(J) THEN 210
180 C=B(I)
190 B(I)=A(J)
200 A(J)=C
210 NEXT J
220 NEXT I
230 B=0
240 FOR I=1 TO N
250 B=B+B(I)
260 NEXT I
270 PRINT " SUMA CELOR ";M;" NUMERE "
275 PRINT " CU VALOAREA CEA MAI MARE: ";
280 PRINT B
290 STOP
300 END
```

Programul solicită prin dialog numărul total de elemente şi valorile lor, apoi solicită valoarea m.

Se afişează suma celor m numere.

10.13. Calculul valorii medii ponderate a unei variabile aleatoare

Se efectuează calculul valorii medii ponderate după formula :

$$V=\frac{\sum_{i=1}^{n} a_i \cdot x_i}{\sum_{i=1}^{n} a_i}.$$

Se introduc prin dialog numărul total de măsurători, valorile fiecărei măsurători şi ponderile cu care se efectuează calculul.

Se afişează valoarea V.

```
5   PRINT " VALOAREA MEDIE MATEMATICA CU PONDERE "
10  PRINT " DATI NUMARUL DE MASURATORI "
15  INPUT N
20  DIM M(N),X(N)
25  PRINT " DATI VALORILE MASURATORILOR "
30  MAT INPUT M
35  PRINT " DATI PONDERILE "
40  MAT INPUT X
45  A=0
50  B=0
55  REM " CALCULUL "
60  FOR I=1 TO N
65  A=A+M(I)*X(I)
70  B=B+X(I)
75  NEXT I
80  PRINT " V = ";A/B
85  STOP
90  END
```

10.14. Calculul valorii medii şi abaterii standard a unei variabile aleatoare

Se determină valoarea medie şi abaterea standard după formulele :

$$V=\frac{1}{n}\sum_{i=1}^{n} X_i$$

$$S^2=\frac{1}{n-1}\sum_{i=1}^{n} (X_i-V)^2.$$

Se introduc prin dialog numărul total de măsurători şi valorile lor. Se afişează valoarea medie şi abaterea standard determinată.

```
5   PRINT " CALCULUL VALORII MEDII SI A ABATERII "
10  PRINT " STANDARD "
15  PRINT " CITE MASURATORI ATI EFECTUAT "
20  INPUT N
25  PRINT " DATI VALORILE MASURATORILOR "
30  MAT INPUT M(N)
35  S=0
45  FOR X=1 TO N
50  S=S+M(X)
55  NEXT X
60  V=S/N
65  S=0
75  FOR X=1 TO N
80  S=S+(M(X)-V)^2
85  NEXT X
90  Z=SQR(S/(N-1))
95  PRINT " VALOAREA MEDIE: ";V
100 PRINT " ABATEREA STANDARD: ";Z
105 STOP
110 END
```

10.15. Tabela valorilor unei funcții definită pe intervale

Programul solicită prin dialog trei intervale ce vor alcătui domeniul de definiție, de forma:

$$[A, B], (B, C), [C, D]$$

Funcția este definită astfel:

- pe intervalul [A, B] în linia 35
- pe intervalul (B, C) în linia 45
- pe intervalul [C, D] în linia 55

```
5  PRINT " FUNCTIA VARIABILA IN INTERVALE "
10 PRINT " DATI INTERVALELE A,B,C,D "
15 INPUT A,B,C,D
20 FOR X=A TO D STEP 0.25
25 K = 4+SGN(X-A)+SGN(X-B)+SGN(X-C)+SGN(X-D)
30 ON K GOTO 35,35,45,45,45,55,55
35 F=0
40 GOTO 60
45 F = SQR(X-B)
50 GOTO 60
55 F = SQR(C-B)+C·X
60 PRINT "X=";X;"F(X)=";F
65 NEXT X
70 STOP
75 END
```

Definițiile pot fi modificate în cadrul programului. Linia 20 definește pasul cu care se efectuează tabelarea valorilor funcției. Ca rezultat, se afișează valorile funcției și ale argumentului pe intervalele date.

10.16. Calculul volumului butoiului

Se determină volumul butoiului pe baza formulei aproximative:

$$V \approx \pi\left(\frac{2D+d}{G}\right)^2 \cdot L.$$

unde :

D este diametrul secţiunii prin vrană

d este diametrul fundului butoiului

L este înălţimea butoiului.

Se solicită prin dialog circumferinţa la vrană şi la fund şi înălţimea, afişînd ca rezultat volumul butoiului.

```
10  PRINT " VOLUMUL BUTOIULUI "
20  PRINT " DATI CIRCUMFERINTA LA VRANA "
30  INPUT C1
40  PRINT " DATI CIRCUMFERINTA LA FUND "
50  INPUT C2
60  PRINT " DATI INALTIMEA "
70  INPUT L
80  D1=C1/PI
90  D2=C2/PI
100 V=PI*L*((2*D1+D2)/6)^2
110 PRINT " VOLUMUL BUTOIULUI: ";V
120 STOP
130 END
```

10.17. Calculul volumului şi suprafeţei torului

Se determină volumul şi suprafaţa torului pe baza formulelor :

$$V = 2\pi^2 R r^2$$

$$A = 4\pi^2 R r$$

unde :

R este raza torului

r este raza secţiunii torului

Se solicită prin dialog valorile R şi r, şi se afişează suprafaţa şi volumul determinate.

```
10  PRINT " VOLUMUL SI SUPRAFATA TORULUI "
20  PRINT " DATI RAZA TORULUI "
30  INPUT R1
40  PRINT " DATI RAZA SECTIUNII "
50  INPUT R2
60  D1=2*R1
70  D2=2*R2
80  F=PI^2*D1*D2
90  V=F*D2/4
100 PRINT " SUPRAFATA: ";F
110 PRINT " VOLUMUL: ";V
120 STOP
130 END
```

10.18. Calculul perimetrului şi suprafeţei unui triunghi

Programul permite calcularea perimetrului şi ariei unui triunghi în două cazuri : fie prin introducerea lungimilor celor trei laturi, fie prin introducerea lungimii unei laturi şi a mărimii celor două unghiuri adiacente. Se afişează perimetrul şi suprafaţa determinată.

```
10  PRINT " CALCULUL PERIMETRULUI SI "
15  PRINT " SUPRAFETEI TRIUNGHIULUI "
20  PRINT " SE CUNOSC LUNGIMILE LATURILOR "
30  PRINT " DATI LUNGIMILE LATURILOR "
40  INPUT X,Y,Z
50  P=X+Y+Z
60  Q=P/2
70  S=SQR(Q*(Q-X)*(Q-Y)*(Q-Z))
80  PRINT " PERIMETRUL: ";P
90  PRINT " SUPRAFATA: ";S
100 STOP
110 END
```

```
10  PRINT " INTR-UN TRIUNGHI SE CUNOSC: "
20  PRINT " LUNGIMEA UNEI LATURI SI ";
30  PRINT " UNGHIURILE PE ACEASTA LATURA "
40  PRINT " DATI LUNGIMEA LATURII "
50  INPUT Y
60  PRINT " DATI UNGHIURILE "
70  INPUT U
80  INPUT W
90  V=360-(U+W)
100 X=Y*SIN(U*PI/180)/SIN(V*PI/180)
110 Z=Y*SIN(W*PI/180)/SIN(V*PI/180)
120 P=X+Y+Z
130 Q=P/2
140 S=SQR(Q*(Q-X)*(Q-Y)*(Q-Z))
150 PRINT " LATURILE: ";X,Y,Z
160 PRINT " UNGHIURILE: ";U,V,W
170 PRINT " PERIMETRUL: ";P
180 PRINT " SUPRAFATA: ";S
190 STOP
200 END
```

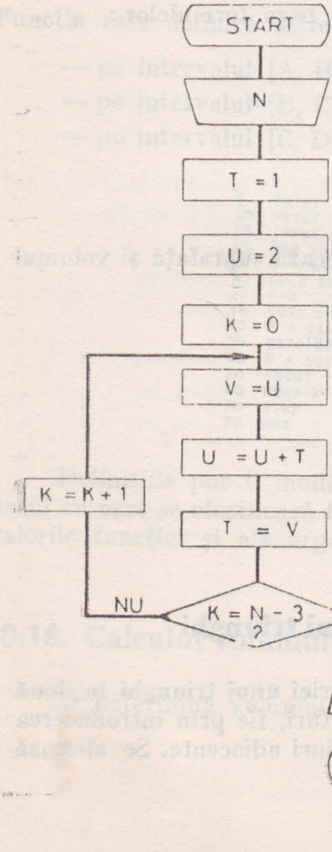
10.19. Calculul celui de-al N-lea număr din şirul lui Fibonacci.

S-a plecat de la relaţia de recurentă :

$$F(N)=F(N-1)+F(N-2),\ (\forall)\, N \geqslant 3$$

$$F(1)=1\ ;\ F(2)=2.$$

Programul se bazează pe organigrama dată în figura 10.5.




```
10  PRINT"PROGRAMUL TIPARESTE"
20  PRINT"AL N-LEA NUMAR DIN"
30  PRINT"SIRUL LUI FIBONACCI"
40  PRINT"N="
50  INPUT N
60  T=1
70  U=2
80  FOR K=0 TO N-3
90  V=U
100 U=U+T
110 T=V
120 NEXT K
130 PRINT"F( ",N," )=",U
140 END
```

Fig. 10.5. Organigrama calculului celui de-al N-lea număr din şirul lui Fibonacci.

10.20. Calculul aproximativ al rădăcinii

$$\sqrt[n]{X}=Z,\ n\geq 2,\ X>\emptyset$$

S-a folosit relaţia de recurenţă :

$$Z_{k+1}=\frac{1}{n}\left[(n-1)\,Z_k+\frac{x}{Zk^{n-1}}\right](\forall)\,K\geqslant\emptyset$$

$$Z_0=\begin{cases}x, & \text{dacă } x>1,\\ 1, & \text{dacă } x\leqslant 1.\end{cases}$$

Programul se bazează pe organigrama dată în figura 10.6.

Procesul de calcul se opreşte cînd este satisfăcută relaţia :

$$|\,Z_{k+1}-Z_k\,|<\varepsilon$$

```
5     PRINT"CALCULUL RADICALULUI"
10    PRINT"DE ORDIN N>=2 DINTR-UN"
15    PRINT"NUMAR POZITIV X"
20    PRINT"N="
25    INPUT N
30    IF N>=2 THEN 45
35    PRINT"EROARE"
40    GOTO 20
45    PRINT"X="
50    INPUT X
55    IF X>0 THEN 85
60    IF X=0 THEN 75
65    PRINT"EROARE"
70    GOTO 45
75    PRINT"REZULTATUL ESTE 0."
80    GOTO 130
85    IF X>1 THEN 100
90    Y=1
95    GOTO 105
100   Y=X
105   Z=((N-1)*Y+X/Y^(N-1))/N
110   IF ABS(Z-Y)<=10^(-5) THEN 125
115   Y=Z
120   GOTO 105
125   PRINT"RADICALUL DIN ",X," ESTE:",Z
130   END
```

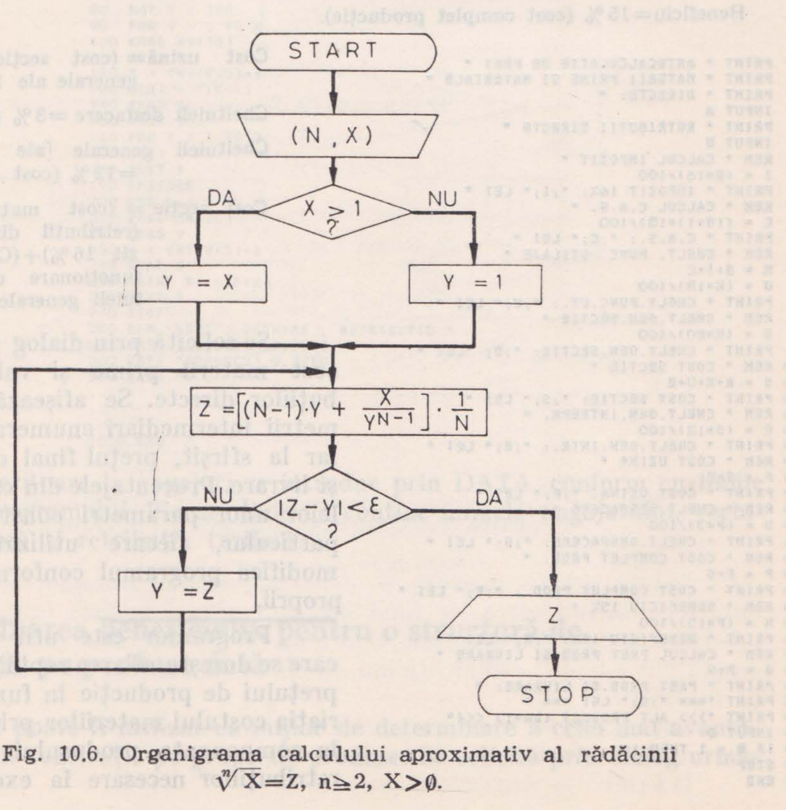


Fig. 10.6. Organigrama calculului aproximativ al rădăcinii $\sqrt[n]{X}=Z,\ n\geq 2,\ X>\emptyset$.

Capitolul 11.

Microcalculatorul aMIC în economie și tehnică

11.1. Antecalculația de preț pentru un produs

Programul efectuează calculul prețului de producție și livrare a unui produs, în baza formulei:

Preț producție și livrare = (cost complet prod.) + (beneficiu)

Cost complet prod. = (cost uzină) + (cheltuieli desfacere)

Beneficiu = 15 % (cost complet producție).

Cost uzină = (cost secție) + (cheltuieli generale ale întreprinderii).

Cheltuieli desfacere = 3 % (cost uzină).

Cheltuieli generale ale întreprinderii = 12 % (cost secție)

Cost secție = (cost materii prime) + (retribuții directe) + (impozit 16 %) + (C.A.S.) + (chelt. funcționare utilaje) + (cheltuieli generale secție).

```
5   PRINT " ANTECALCULATIE DE PRET "
10  PRINT " MATERII PRIME SI MATERIALE "
15  PRINT " DIRECTE: "
20  INPUT A
25  PRINT " RETRIBUTII DIRECTE "
30  INPUT B
35  REM " CALCUL IMPOZIT "
40  I = (B*16)/100
45  PRINT " IMPOZIT 16%: ";I;" LEI "
50  REM " CALCUL C.A.S. "
55  C = ((B+I)*15)/100
60  PRINT " C.A.S.: ";C;" LEI "
65  REM " CHELT. FUNC  UTILAJE "
70  K = B+I+C
75  U = (K*18)/100
80  PRINT " CHELT.FUNC.UT.: ";U;" LEI "
85  REM " CHELT.GEN.SECTIE "
90  D = (K*80)/100
95  PRINT " CHELT.GEN.SECTIE: ";D; "LEI "
100 REM " COST SECTIE "
105 S = A+K+U+D
110 PRINT " COST SECTIE: ";S;" LEI "
115 REM " CHELT.GEN.INTREPR. "
120 E = (S*12)/100
125 PRINT " CHELT.GEN.INTR.: ";E;" LEI "
130 REM " COST UZINA "
135 F = S+E
140 PRINT " COST UZINA: ";F;" LEI "
145 REM " CHELT.DESFACERE "
150 G = (F*3)/100
155 PRINT " CHELT.DESFACERE: ";G;" LEI "
160 REM " COST COMPLET PROD. "
165 P = F+G
170 PRINT " COST COMPLET PROD.: ";P;" LEI "
175 REM " BENEFICIU 15% "
180 N = (P*15)/100
185 PRINT " BENEFICIU 15%: ";N;" LEI "
190 REM " CALCUL PRET PROD.SI LIVRARE "
195 Q = P+N
200 PRINT " PRET PROD.SI LIVRARE: "
205 PRINT "*** ";Q;" LEI *** "
210 PRINT ">>> ALT PRODUS? (DA=1) <<<"
215 INPUT D
220 IF D = 1 THEN 10
225 STOP
230 END
```

Se solicită prin dialog variabilele: cost materii prime și valoarea retribuțiilor directe. Se afișează toți parametrii intermediari enumerați mai sus, iar la sfîrșit, prețul final de producție și livrare. Procentajele din cadrul calculelor unor parametri constituie un caz particular, fiecare utilizator putînd modifica programul conform cerințelor proprii.

Programul este util în cazul în care se dorește aflarea rapidă a variației prețului de producție în funcție de variația costului materiilor prime ce intră în componența produsului și a valorii retribuțiilor necesare la executarea sa.

11.2. Calculul primei acordate după grupa de vechime

Se consideră următoarele grupe de vechime:

I. < 5 ani — IV. 15—2Ø ani

II. 5—1Ø ani — V. > 2Ø ani

III. 1Ø—15 ani

Programul solicită prin dialog totalul sumei ce se va acorda drept prime pentru fiecare grupă de vechime. Se determină suma ce revine fiecărui angajat conform cu grupa din care face parte, și afișează lista tuturor angajaților, însoțită de retribuția tarifară și prima acordată.

```
10  PRINT " CALCULUL PRIMEI IN FUNCTIE "
15  PRINT " DE GRUPA DE VECHIME "
20  PRINT " GRUPE VECHIME  < 5   5-10   "
25  PRINT " 10-15   15-20 ,  > 20 ANI "
30  DIM P(5),N$(10),T(5)
40  PRINT " INTROD  SUMA/GRUPA CE SE "
45  PRINT " ACORDA DREPT PRIMA "
50  MAT INPUT P
60  PRINT " INTROD.NR TOTAL ANGAJATI "
70  INPUT M
80  MAT T = ZER
90  FOR I = 1 TO M
100 READ N$(10)
110 READ V
120 K = INT(V/5)+1
130 T(K) = T(K)+1
140 READ S
150 NEXT I
160 FOR I = 1 TO 5
170 P(I) = P(I)/T(I)
180 NEXT I
190 RESTORE
200 FOR I = 1 TO M
210 READ N$(10)
220 READ V
230 K = INT(V/5)+1
240 READ S
250 PRINT N$;S;P(K)
260 NEXT I
270 STOP
280 REM "NUME , VECHIME , RETRIBUTIE "
290 DATA "IONESCU",3,3200
300 DATA "POPESCU",8,2710
310 DATA "...
.
.
.
1000 END
```

Datele referitoare la angajați se introduc prin DATA conform cu liniile de la sfîrșitul programului. Fiecare linie va conține numele angajatului, vechimea sa în muncă și retribuția tarifară.

11.3. Determinarea beneficiului pentru o structură de fabricație pe produse dată

Programul poate fi utilizat ca mijloc de determinare a celei mai avantajoase structuri de fabricație pe grupe de produse. Se solicită prin dialog următoarele date:

— numărul şi numele produselor de fabricaţie ;

— numărul şi numele materialelor (materiilor prime) necesare fabricării tuturor produselor ;

— pentru fiecare produs, cantitatea de materii prime ce intră în fabricaţia sa ;

— pentru fiecare materie primă, cantitatea existentă în stoc ;

— pentru fiecare produs, beneficiul estimat ;

— structura de fabricaţie (cîte bucăţi se vor fabrica din fiecare produs).

Se verifică mai întîi dacă în baza datelor introduse, se poate realiza structura de fabricaţie cerută (nedepăşirea stocului de materii prime), iar dacă rezultatul verificării este pozitiv, se afişează beneficiul total pentru această structură. Se poate repeta întregul proces, modificînd doar structura de fabricaţie, pînă la determinarea unei structuri optimizate (fig. 11.1).

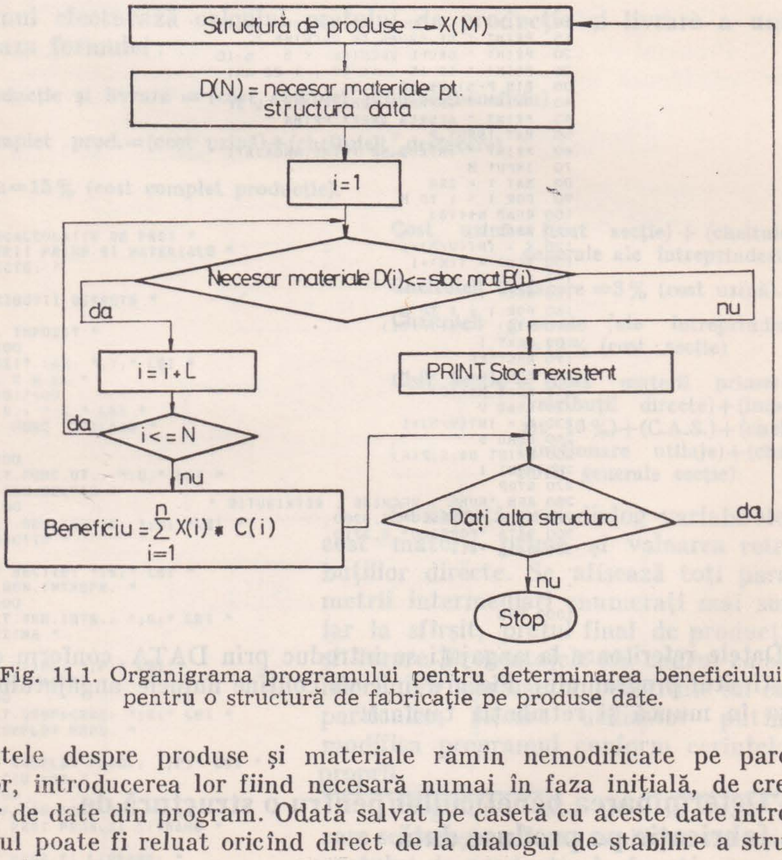


Fig. 11.1. Organigrama programului pentru determinarea beneficiului pentru o structură de fabricaţie pe produse date.

Datele despre produse şi materiale rămîn nemodificate pe parcursul iteraţiilor, introducerea lor fiind necesară numai în faza iniţială, de creare a tabelelor de date din program. Odată salvat pe casetă cu aceste date introduse, programul poate fi reluat oricînd direct de la dialogul de stabilire a structurii de fabricaţie.

Variabilele M, N conţin numărul de produse respectiv de materiale. Matricile utilizate conţin următoarele date:

D $ (M) — denumirile produselor

P $ (N) — denumirile materialelor

A(M, N) — cîte unităţi din materialul N necesită produsul M

B (N) — stocul de materiale existente

C (M) — beneficiul estimat pe produse

Ciclul de verificare începe de la linia 300 :

```
10 PRINT " DETERMINAREA BENEFICIULUI "
12 PRINT " IN FUNCTIE DE O STRUCTURA "
15 PRINT " DE FABRICATIE ALEASA "
20 PRINT " INTROD.NR.TOTAL PRODUSE "
25 PRINT " SI NR.TOTAL MATERIALE "
30 INPUT M,N
40 DIM A(M,N),X(N),B(N),C(N),D(N)
50 DIM D$(M,10)
60 DIM P$(N,10)
70 PRINT " INTROD.DENUMIRI PRODUSE "
80 FOR I = 1 TO M
90 INPUT D$(I)
100 NEXT I
110 PRINT " INTROD.DENUMIRI MATERIALE "
120 FOR I = 1 TO N
130 INPUT P$(I)
140 NEXT I
145 PRINT " INTROD CONSUM MAT/PROD. "
150 FOR I = 1 TO M
160 FOR J = 1 TO N
170 PRINT D$(I);" NR.BUC.DIN: ";P$(J)
180 INPUT A(I,J)
190 NEXT J
200 NEXT I
205 PRINT " INTROD. STOC MAT "
210 FOR J = 1 TO N
220 PRINT " DIN ";P$(J);" CITE U/M AVETI "
230 INPUT B(J)
240 NEXT J
245 PRINT " INTROD BENEF.ESTIMAT
250 FOR I = 1 TO M
260 PRINT D$(I);" CE BENEF.REALIZEAZA "
270 INPUT C(I)
280 NEXT I
290 PRINT " INTROD.STRUCTURA DE FABR. "
295 PRINT " NR. PRODUSE PE TIPURI "
300 FOR I = 1 TO M
310 PRINT D$(I);" PRODUS "
320 INPUT X(I)
330 NEXT I
340 MAT D = ZER
350 FOR I = 1 TO N
360 FOR J = 1 TO M
370 D(I) = D(I)+X(J)*A(J,I)
380 NEXT J
390 NEXT I
400 J = 0
410 FOR I = 1 TO N
420 IF J <> 0 THEN 460
430 IF D(I) <= B(I) THEN 460
440 J = 1
450 PRINT " STRUCTURA DATA NU SE POATE "
455 PRINT " REALIZA "
460 NEXT I
470 IF J = 0 THEN 520
480 PRINT " ALTA STRUCTURA? (DA=1) "
490 INPUT E
500 IF E = 1 THEN 290
510 STOP
520 S = 0
530 FOR I = 1 TO M
540 S = S+X(I)*C(I)
550 NEXT I
560 PRINT " BENEFICIUL ESTE: ";S
570 GOTO 480
580 END
```

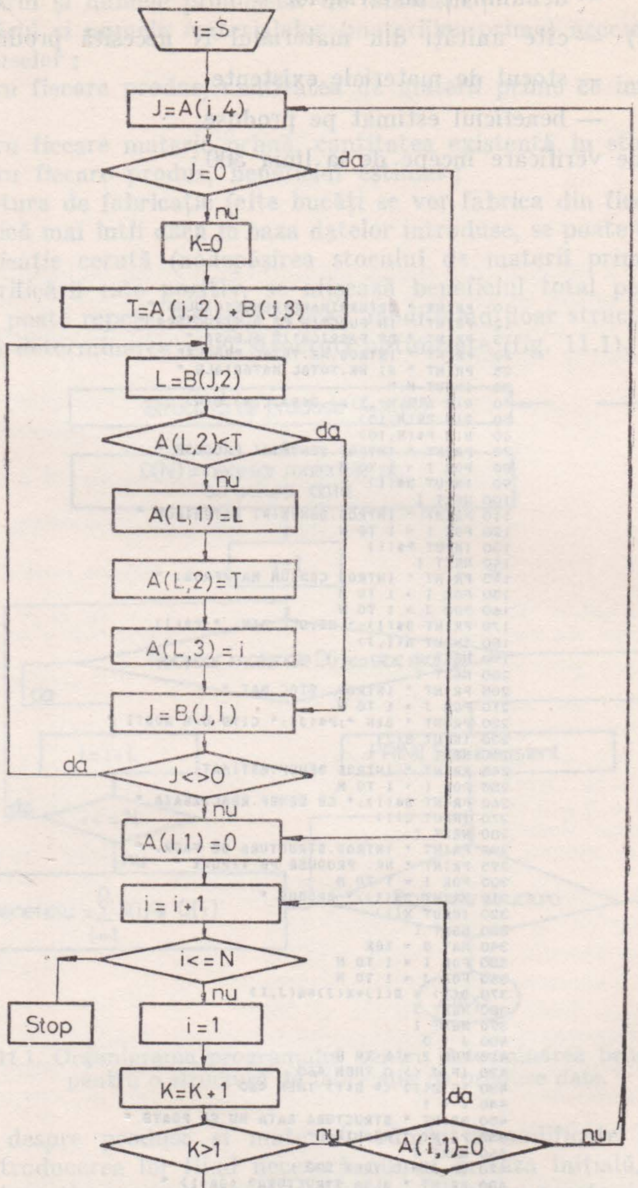


Fig. 11.2. Organigrama programului pentru determinarea drumului minim între două noduri ale unui graf dat.

11.4. Determinarea drumului minim între două noduri ale unui graf dat

Programul consideră un graf ca fiind definit prin noduri şi arce, fiecare nod fiind etichetat cu un număr întreg pozitiv. Se solicită prin dialog :

— numărul total de noduri = N
— numărul total de arce = E
— eticheta (numărul) nodului de pornire (primul nod din drumul de minimizat) = S
— eticheta (numărul) nodului de sosire (ultimul nod din drum) = H
— pentru fiecare arc :
  — nodul iniţial
  — nodul final
  — lungimea sa.

Din datele introduse se completează masivul A (N, 4) în care A (L, 2) = =1 000 000 considerat un timp foarte mare respectiv A (S, 2) = 0 pentru nodul de pornire. Masivul B (E, 3) se completează cu numărul nodurilor pe arce şi cu timpii necesari parcurgerii arcelor. Conectorul A (L, 4) va conţine legătura către liniile din masivul B.

În A (L, 2) se totalizează timpii necesari parcurgerii grafului, conform schemei din figura 11.2.

Se afişează sub forma unui şir de etichete (numere) drumul minim între cele două noduri specificate. Afişarea se face în ordinea inversă parcurgerii sale (de la nodul de destinaţie la nodul de pornire).

Programul poate fi utilizat într-o serie de domenii unde este necesară optimizarea distanţelor parcurse, planificarea activităţilor de producţie etc. Se va considera lungimea unui arc drept măsură pentru distanţe de parcurs, timpi necesari în efectuarea unor operaţii tehnologice etc.

```
10 PRINT " DRUMUL CRITIC "
20 PRINT " NUMAR DE NODURI "
30 INPUT N
40 PRINT " NUMAR DE ARCE "
50 INPUT E
60 DIM A(N,4),B(E,3)
70 PRINT " NR.-UL NODULUI DE PORNIRE "
80 INPUT S
90 PRINT " NR.-UL NODULUI DE SOSIRE "
100 INPUT H
110 FOR I = 1 TO N
120 A(I,1) = 1
130 A(I,2) = 1000000
140 A(I,3) = O
150 A(I,4) = O
160 NEXT I
165 REM " TIMPUL DE PORNIRE = O "
170 A(S,2) = O
180 PRINT " NOD 1 , NOD 2   LUNGIME "
190 FOR P = 1 TO E
200 INPUT A,B,R
210 B(P,2) = B
220 B(P,3) = R
225 REM " INLANTUIRE TABLOU A SI B
230 B(P,1) = A(A,4)
240 A(A,4) = P
250 NEXT P
260 I = S
270 J = A(I,4)
280 IF J = O THEN 380
290 K = O
300 T = A(I,2)+B(J,3)
310 L = B(J,2)
320 IF A(L,2) < T THEN 360
330 A(L,1) = 1
340 A(L,2) = T
350 A(L,3) = I
360 J = B(J,1)
370 IF J <> O THEN 300
380 A(I,1) = O
390 I = I+1
400 IF I <= N THEN 450
420 I = 1
430 K = K+1
440 IF K > 1 THEN 470
450 IF A(I,1) = O THEN 390
460 GOTO 270
470 PRINT ""DIN ";S;" IN ";H;" DRUMUL ";
480 PRINT " CEL MAI SCURT ESTE: "
490 PRINT " PRIN NODURILE: ";H;
500 X = A(H,3)
510 IF X = O THEN 550
520 PRINT " , ";X,
530 X = A(X,3)
540 GOTO 510
550 STOP
560 END
```

11.5. Gestiunea unui stoc de magazie de tehnică dentară

Se consideră că există o magazie care primeşte materiale specifice dentare, şi le distribuie la mesele de lucru.

Fiecare masă de lucru elaborează o anumită lucrare, pentru care se consumă o anumită cantitate de materiale.

Fiecărei lucrări îi este asociat un barem de consumuri utilizat pentru verificarea încadrării în normele stabilite.

Programul oferă următoarele opţiuni de lucru:

— punerea la zi a magaziei centrale (primire/eliberare materiale);
— vizualizarea stocului existent în magazia centrală;
— punerea la zi a stocului de materiale la mesele de lucru, cu verificarea consumului de materiale;
— vizualizarea stocului de materiale existent la mesele de lucru.

```
10 PRINT " MAGAZIA DE TEHNICA DENTARA "
20 PRINT " 1= SE TINE LA ZI MAGAZIA CENTRALA "
30 PRINT " 2= VIZUALIZARE STOC MAGAZIE CENTRALA "
40 PRINT " 3= VIZUALIZARE STOC LA MESE "
50 PRINT " 4= SE TINE LA ZI STOCUL LA MESE "
55 PRINT " 5= STOP "
60 PRINT " DATI NUMARUL PRELUCRARII "
70 INPUT X
80 ON X GOTO 90,400,780,880,1540
90 PRINT " SE TINE LA ZI MAGAZIA CENTRALA"
100 PRINT " PRIMITI SAU ELIBERATI MAT.(P/E) "
110 INPUT A$
120 IF A$ = "E" THEN 450
130 IF A$ <> "P" THEN 90
140 PRINT " DATI MATERIALUL SI CANTITATEA "
150 INPUT B$(TO),X
160 Y = 0
170 FOR I = 1 TO N
180 IF B$ <> M$(I) THEN 220
190 O(I) = O(I)+X
200 Y = 1
210 I = N
220 NEXT I
230 IF Y <> 0 THEN 340
240 PRINT " MATERIAL NOU? (D/N)
250 INPUT A$
260 IF A$ = "D" THEN 280
270 GOTO 340
275 REM " SE ADAUGA UN MATERIAL NOU "
280 N = N+1
290 O(N) = X
300 M$(N) = B$
310 FOR I = 1 TO M
320 PRINT " CONSUM SPECIFIC PRODUS ";P$(I);
325 PRINT " DIN MATERIAL ";B$
330 INPUT P(I,N)
335 NEXT I
340 PRINT " MAI PRIMITI MATERIAL (D/N)"
350 INPUT A$
360 IF A$ = "D" THEN 140
370 PRINT " DORITI VIZUALIZARE STOC (D/N) "
380 INPUT A$
390 IF A$ <> "D" THEN 10
400 PRINT " VIZUALIZARE STOC MAGAZIE CENTRALA
410 FOR I = 1 TO N
420 PRINT M$(I),"     ",O(I)
430 NEXT I
440 GOTO 10
450 PRINT " SE ELIBEREAZA MATERIAL LA MESE "
460 PRINT " DATI NUMARUL MESEI "
470 INPUT S
480 IF S <= 0 THEN 460
490 IF S > Q THEN 460
500 PRINT " DATI MATERIALUL SI CANTITATEA "
510 INPUT B$(TO),X
520 Y = 0
530 FOR I = 1 TO N
540 IF B$ <> M$(I) THEN 570
```

```
550 Y = 1
560 I = N
570 NEXT I
580 IF Y <> 0 THEN 610
590 PRINT "   NU EXISTA MATERIALUL IN MAGAZIE "
600 GOTO 680
610 O(Y) = O(Y)-X
620 IF O(Y) >=0 THEN 660
630 PRINT " NU EXISTA CANTITATEA IN MAGAZIE "
640 O(Y) = O(Y)+X
650 GOTO 680
660 REM " SE ADUNA LA STOCUL MESEI "
670 A(S,Y) = A(S,Y)+X
680 PRINT " MAI ELIBERATI LA MASA ";S;" (D/N) "
690 INPUT A$
700 IF A$ = "D" THEN 500
710 PRINT " ELIBERATI LA ALTA MASA? (D/N) "
720 INPUT A$
730 IF A$ = "D" THEN 460
740 PRINT " VRETI VIZUALIZARE STOC MASA (D/N)
750 INPUT A$
760 IF A$ = "D" THEN 780
770 GOTO 10
780 PRINT " VIZUALIZARE STOC MASA "
790 PRINT " DATI NUMARUL MESEI "
800 INPUT S
810 IF S <= 0 THEN 790
820 IF S > Q THEN 790
830 FOR I = 1 TO N
840 PRINT " M$(I),"     ",A(S,I)
850 NEXT I
860 PRINT " MAI VIZUALIZATI STOC LA MASA (D/N) "
870 GOTO 750
880 PRINT " SE DAU REALIZARILE LA MESE "
890 PRINT   DATI NUMARUL MESEI "
900 INPUT S
910 IF S <= 0 THEN 890
920 IF S > Q THEN 890
925 MAT B(M) = ZER
930 PRINT " DATI PRODUSUL SI NR.PROD.REALIZATE "
940 INPUT B$(TO),X
950 Y = 0
960 FOR I = 1 TO M
970 IF B$ <> P$(I) THEN 1010
980 Y = 1
990 B(I) = X
1000 I = M
1010 NEXT I
1020 IF Y <> 0 THEN 1130
1030 PRINT " PRODUS NOU? (D/N) "
1040 INPUT A$
1050 IF A$ <> "D" THEN 1130
1055 REM " SE ADAUGA UN PRODUS NOU "
1060 M = M+1
1070 P$(M) = B$
1080 PRINT " DATI CONSUMURILE SPECIFICE PROD.NOU "
1090 FOR I = 1 TO N
1100 PRINT M$(I);
```

```
1110 INPUT P(M,I)
1120 NEXT I
1130 PRINT " MAI DATI REALIZARE? (D/N)
1140 INPUT A$
1150 IF A$ = "D" THEN 930
1155 MAT C(N) ZER
1160 PRINT " DATI CONSUM MATERIALE LA MASA ";S
1180 INPUT B$(TO),X
1190 Y = 0
1200 FOR I = 1 TO N
1210 IF B$ <> M$(I) THEN 1270
1220 Y = 1
1230 IF A(P,I) < X THEN 1260
1240 Y = 2
1250 C(I) = X
1260 I = N
1270 NEXT I
1280 IF Y = 2 THEN 1340
1290 IF Y = 1 THEN 1320
1300 PRINT " MATERIAL INEXISTENT "
1310 GOTO 1340
1320 PRINT " STOC INSUFICIENT "
1340 PRINT " MAI DATI CONSUM? (D/N) "
1350 INPUT A$
1360 IF A$ = "D" THEN 1170
1370 FOR I = 1 TO N
1380 A(S,I) = A(S,I)-C(I)
1390 NEXT I
1400 MAT D = ZER
1410 FOR I = 1 TO M
1420 FOR J = 1 TO N
1430 D(J) = D(J)+B(I)*P(J,I)
1440 NEXT J
1450 NEXT I
1460 Y = 0
1470 FOR I = 1 TO N
1475 IF D(I) <= C(I) THEN 1500
1480 PRINT " S-A DEPASIT CONSUMUL LA ";M$(I);
1485 PRINT " MATERIAL CU ";C(I)-D(I)
1490 Y = 1
1500 NEXT I
1510 IF Y = 1 THEN 10
1520 PRINT " VATI INCADRAT IN CONSUM SPECIFIC "
1530 GOTO 10
1540 STOP
1600 REM " CREARE FISIER STOC "
1610 REM " SE CREEAZA LA MAXIM 100 MATERIALE "
1615 REM " 20 PRODUSE SI 5 MESE "
1620 DIM O(100),B(20),C(100),D(100)
1630 DIM A(5,100),P(20,100)
1540 DIM P$(20,10),M$(100,10)
1650 REM " SE INTRODUC 5 MATERIALE 2 PRODUSE "
1655 REM " RESTUL SF INTRODUC PRIN PROGRAM "
1660 N = 5
1670 M = 2
1680 Q = 5
1690 MAT O = ZER
1700 P$(1) = "PLAN INCL."
1710 P$(2) = "MONOBLOC"
1720 M$(1) = "CEARA"
1730 M$(2) = "MASA AMBL."
1740 M$(3) = "ALIAJ INOX"
1750 M$(4) = "GIPS"
1760 MAT A = ZER
1770 MAT P = ZER
1780 REM " CONSUMURILE SPECIFICE "
1790 FOR I = 1 TO M
1800 FOR J = 1 TO N
1810 READ P(I,J)
1820 NEXT J
1830 NEXT I
1835 REM " CONSUM SPECIFIC DUPA BAREM "
1840 DATA 5,10,14,600,60
1850 DATA 15,0,0,200,60
1900 GOTO 10
2000 END
```

Programul poate fi utilizat ca model de referință pentru gestiunea stocurilor la magazii mici-medii, spațiul de memorie ocupat de datele referitoare la magazie depinzînd în mare măsură de modul în care sînt codificate materialele din stoc. Asocierea unui număr mare de date unui material dat va reduce numărul total de materiale ce pot fi cuprinse în tabelele de descriere a magaziei. Implicit, numărul maxim de materiale diferite nu poate fi mai mare de 254, deoarece interpretorul BASIC nu admite tablouri a căror dimensiune să fie mai mare de 254.

Pentru prima lansare se utilizează comanda RUN 1600 care dimensionează matricile utilizate. Prin inițializare s-au introdus 5 materiale, 2 produse și consumurile specifice aferente. Prin program se mai introduc restul de materiale și produse. După inițializare programul se lansează cu comanda GOTO10 pentru a nu inițializa masivele completate cu stocul magaziei centrale, respectiv cu stocul meselor de lucru.

11.6. Balanța de verificare debit-credit

Programul constituie un exemplu simplu de realizare a unei balanțe de verificare. La prima rulare, programul solicită introducerea numărului total de conturi și a numerelor de cont pentru care se vor face operațiile de debit/credit. Se introduc apoi, pentru fiecare număr de cont, debitele și creditele pe luna în curs. Se afișează apoi balanta de verificare pe subconturi ; la umplerea ecranului, afișajul se oprește, putînd fi reluat prin apăsarea unei taste oarecare. După afișarea balanței pe subconturi, se afișează balanta de verificare pe conturi, împreună cu totalurile debitoare, creditoare și pe conturi.

Rulările următoare nu mai solicită introducerea numerelor de cont, ci doar debitul și creditul pe luna curentă. Verificările se fac începînd cu soldul lunii precedente. Pentru a nu distruge masivul cu soldul lunii precedente, lansarea ulterioară a programului se realizează cu comanda GOTO 100.

```
10  PRINT " BALANTA DE VERIFICARE "
20  PRINT " DATI NR.TOTAL DE CONTURI "
30  INPUT N
40  DIM K(N),C(N),D(N)
50  DIM R(N),B(N),S(N)
60  MAT C = ZER
70  MAT D = ZER
80  PRINT " INTRODUCETI CONTURILE "
90  MAT INPUT K
100 PRINT " INTRODUCETI CREDIT/DEBIT "
105 PRINT " LUNA CURENTA "
110 FOR I = 1 TO N
120 PRINT K(I);"?"
130 PRINT "DB:";
140 INPUT B(I)
150 PRINT "CR:";
160 INPUT R(I)
170 NEXT I
```

```
180 FOR I = 1 TO N
190 C(I) = C(I)+R(I)
200 D(I) = D(I)+B(I)
210 S(I) = D(I)-C(I)
220 NEXT I
230 INIT P
240 PRINT AT(1,1);" BALANTA DE VERIFICARE
245 B$ = INKEY$
250 U = 3
260 FOR I = 1 TO N
270 PRINT AT(U,1);K(I),S(I)
280 U = U+1
290 IF U < 31 THEN 350
300 A$ = INKEY$
310 IF A$ = B$ THEN 300
320 U = 3
330 INIT P
340 PRINT AT(1.1);" BALANTA DE VERIFICARE
350 NEXT I
360 A$ = INKEY$
370 IF A$ = B$ THEN 360
380 T = S(1)
390 M = INT(K(1)/1000)
400 INIT P
410 PRINT AT(1,1)," TOTAL CONT
420 U = 3
430 FOR I = 2 TO N
440 IF M <> INT(K(I)/1000) THEN 470
450 T = T+S(I)
460 GOTO 570
470 PRINT AT(U,1);M,T
480 M   INT(K(I)/1000)
490 T = S(I)
500 U = U+1
510 IF U < 31 THEN 570
520 A$ = INKEY$
530 IF A$ = B$ THEN 520
540 U = 3
550 INIT P
560 PRINT AT(1,1);" TOTAL CONT "
570 NEXT I
580 PRINT AT(U,1);M,T
610 A$   INKEY$
620 IF A$ = B$ THEN 610
630 INIT P
640 PRINT AT(1,1);" TOTAL BALANTA "
650 M = 0
660 U = 0
670 0 = 0
680 FOR I = 1 TO N
690 M = M+C(I)
700 U = U+D(I)
710 0 = 0+S(I)
720 NEXT I
730 PRINT AT(3,1);"TOTAL DB: ";U
740 PRINT AT(5,1);"TOTAL CR: ";M
750 PRINT AT(7,1):"BALANTA : ";0
760 STOP
```

11.7. Transformarea stea-triunghi şi reciproc

Programul determină rezistenţele echivalente celor două transformări după formulele :

a) (transformarea stea-triunghi)

$$\begin{cases} r_{12} = r_1 + r_2 + \dfrac{r_1 \cdot r_2}{r_3} \\ r_{23} = r_2 + r_3 + \dfrac{r_2 \cdot r_3}{r_1} \\ r_{31} = r_3 + r_1 + \dfrac{r_3 \cdot r_1}{r_2} \end{cases}$$

b) $r_i = \frac{r_i k - r_{ji}}{r_{ij} + r_{jk} + r_{ki}}$ (transformarea triunghi-stea)

```
10  PRINT " CALCUL REZISTENTE ECHIVALENTE "
20  PRINT " 1= SCHEMA ELECTRICA STEA IN TRIUNGHI "
25  PRINT " 2= SCHEMA ELECTRICA TRIUNGHI IN STEA "
30  INPUT X
40  IF X = 1 THEN 210
50  PRINT " SCHEMA ELECTRICA TRIUNGHI IN STEA "
60  PRINT " DATI REZISTENTELE IN TRIUNGHI "
70  PRINT " R12 = ";
80  INPUT A
90  PRINT " R23 = ";
100 INPUT B
110 PRINT " R31 = ";
120 INPUT C
130 S = A+B+C
140 D = (A*C)/S
150 E = (B*A)/S
160 F = (C*B)/S
170 PRINT " R1 = ";D
180 PRINT " R2 = ";E
190 PRINT " R3 = ";F
200 STOP
210 PRINT " SHEMA ELECTRICA STEA IN TRIUNGHI "
215 PRINT " DATI REZISTENTELE IN STEA "
220 PRINT " R1 = ";
230 INPUT D
240 PRINT " R2 = ";
250 INPUT E
260 PRINT " R3 = ";
270 INPUT F
280 A = D+E+(D*E)/F
290 B = E+F+(E*F)/D
300 C = F+D+(F*D)/E
310 PRINT " R12 = ";A
320 PRINT " R23 = ";B
330 PRINT " R31 = ";C
340 STOP
350 END
```

Se solicită prin dialog tipul transformării și valorile rezistențelor corespunzătoare. Se afișează valorile rezistențelor echivalente.

11.8. Dimensionarea liniilor de alimentare în curent continuu

Se consideră un conductor de cupru cuplat la o sursă de curent continuu. De la acest conductor sînt alimentați mai mulți consumatori, fiecare avînd un consum Pi și o distanță Li de la sursa de alimentare. Secțiunea conductorului se deduce pe baza formulei :

$$S = \frac{2 \cdot 100}{\gamma \Delta U \% U^2} \sum PL,$$

unde :

$\gamma = \frac{1}{\rho}$ este conductivitatea conductorului

$\Delta U\%$ este căderea de tensiune maxim admisă

U este tensiunea nominală

P este puterea absorbită de fiecare consumator

L este distanța consumatorului față de sursa de alimentare.

```
10  PRINT " CALCULUL LINIEI CONDUCTOARE "
15  PRINT " DE CURENT CONTINUU SUB "
20  PRINT " TENSIUNE DE 220 V,DIN CUPRU "
30  PRINT " DATI NUMARUL DE CONSUMATORI "
40  INPUT N
50  PRINT " DATI DISTANTELE DE LA SURSA "
55  PRINT " IN METRI "
60  DIM L(N)
70  MAT INPUT L
80  PRINT " DATI CONSUMURILE "
85  PRINT " IN WATI "
90  DIM C(N)
100 MAT INPUT C
110 S = O
120 FOR I = 1 TO N
130 S = S+C(I)*L(I)
140 NEXT I
150 S = S*2*100/(53*5*220^2)
160 PRINT " S = ";S;" MM^2 "
170 STOP
180 END
```

Se solicită prin dialog numărul de consumatori, distanțele de sursă și consumurile fiecăruia. Se afișează valoarea secțiunii, considerînd sursa de alimentare de 22Ø Vc.c. și prinderea maximă de tensiune $\leqslant$ 3% din tensiunea nominală.

11.9. Determinarea greutății materialelor ce intră în componența unui corp eterogen

Se consideră un corp compus din două materiale. Fie G greutatea acestui corp, cîntărit în aer, și Ga greutatea sa, cîntărit în apă. Fie ρ_a densitatea apei, ρ1 densitatea primului material și ρ2 densitatea celui de-al doilea material. Rezultă :

$$G1 = \frac{\rho_1\rho_2\,(G-G_a)-G\rho_a\rho_1}{\rho_a\,(\rho_2-\rho_1)} \text{ greutatea primului material}$$

$$G2 = \frac{\rho_1\rho_2\,(G-G_a)-G\rho_a\rho_2}{\rho_a\,(\rho_1-\rho_2)} \text{ greutatea celui de-al doilea material.}$$

Se introduc prin dialog valorile celor două greutăți, precum și densitățile fiecărei componente. Se afișează ca rezultat greutatea fiecărei componente.

```
10  PRINT " GREUTATEA MATERIALELOR COMPONENTE "
20  PRINT " ALE UNUI CORP HETEROGEN "
30  PRINT " DATI GREUTATEA CORPULUI IN AER "
40  INPUT G1
50  PRINT " DATI GREUTATEA CORPULUI IN APA "
60  INPUT G2
70  PRINT " DATI DENSITATEA MAT. COMPONENTE "
80  INPUT R1,R2
90  REM " DENSITATEA APEI = 1 "
100 R3 = 1
110 V1 = (R1*R2*(G1-G2)-G1*R3*R1)/(R3*(R2-R1))
120 V2 = (R1*R2*(G1-G2)=G1*R3*R2)/(R3*(R1-R2))
130 PRINT " CORPUL DE ";G1;" GREUTATE "
140 PRINT " CONTINE ";V1;" DIN MATERIAL ";R1;
145 PRINT " DENSITATE "
150 PRINT " CONTINE ";V2, DIN MATERIAL ";R2;
155 PRINT   DENSITATE "
160 STOP
170 END
```

11.10. Dimensionarea grinzilor de beton armat

Programul determină înălţimea utilă şi aria armăturii unei grinzi de beton armat, pe baza formulelor:

$$h_0=\frac{\sqrt{\frac{m}{1\cdot r_b}}}{\sqrt{\zeta\left(1-\frac{\zeta}{2}\right)}}$$

$$a=\frac{m}{r_a\cdot h_0\cdot\left(1-\frac{\zeta}{2}\right)}$$

unde:

h_0 este înălţimea utilă;
a este aria armăturii;
m este momentul de încovoiere
l este lăţimea grinzii
ra este rezistenţa armăturii
rb este rezistenţa betonului
ξ este factor limită=0,6

Se introduc prin dialog valorile pantru ra, rb, l şi m. Se afişează valorile înălţimii utile şi ale ariei armăturii.

```
10  PRINT " DIMENSIONARE GRINDA DE BETON ARMAT "
20  REM " SE CONSIDERA C S I LIMITA = 0.6 "
30  P = 0.6
40  PRINT " REZISTENTA ARMATURII IN KGF/CM^2 "
50  INPUT A
60  PRINT " REZISTENTA BETONULUI IN KGF/CM^2 "
70  INPUT C
80  PRINT " LATIMEA GRINZII IN CM "
90  INPUT B
100 PRINT " MOMENTUL INCOVOIETOR IN KGF.CM "
110 INPUT M
120 T = 1-P/2
130 S = P*T
140 R = 1/SQR(S)
150 H = R*SQR(M/(B*C))
160 Z = M/(A*H*T)
170 REM " INALTIMEA UTILA "
180 PRINT "HO = ";H;" CM "
190 PRINT "ARIA ARMATURII = ";Z;" CM^2 "
200 STOP
210 END
```

11.11. Calculul secţiunii elementelor de construcţie

Programul determină caracteristicile principale ale diafragmelor utilizate în construcţii, pornind de la descompunerea secţiunii diafragmei în dreptunghiuri elementare. Se solicită ca date iniţiale numărul de dreptunghiuri elementare ce compun secţiunea, precum şi dimensiunile unui asemenea dreptunghi. Se afişează drept rezultat:

— aria secţiunii diafragmei;
— momentul de inerţie;
— coordonatele centrului de greutate.

```
10 PRINT " CALCULUL SECTIUNII ELEMENTELOR "
15 PRINT " DE CONSTRUCTII "
20 PRINT " DATI NR.DREPTUNGHIURI ELEMENTARE "
30 INPUT N
40 DIM B(N),H(N)
50 PRINT " DATI LATIMEA,INALTIMEA IN CM "
60 FOR J = 1 TO N
70 INPUT B(J)
80 INPUT H(J)
90 NEXT J
100 D = 0
110 H = 0
120 A = 0
130 I = 0
140 FOR J = 1 TO N
150 K = B(J)*H(J)
160 L = H+H(J)/2
170 D = (A*D+K*L)/(A+K)
180 A = A+K
190 I = I+(H(J)^3*B(J))/12
200 H = H(J)+H
210 NEXT J
220 H = 0
230 FOR J = 1 TO N
240 L = H+H(J)/2
250 I = I+B(J)*H(J)*(L-D)^2
260 H = H+H(J)
270 NEXT J
280 PRINT "ARIA = ";A;" CM^2 "
290 PRINT "MOMENT INERTIE = ";I;" CM^4 "
300 PRINT "HS = ";D;" CM "
310 PRINT "HD = ";H-D;" CM "
320 STOP
330 END
```

11.12. Determinarea momentelor de încastrare perfectă ale unei grinzi de beton armat

Se consideră o grindă de beton supusă atît unei forțe uniform distribuite, cît şi unui număr oarecare de forțe concentrate. Se determină valoarea momentului de încastrare stînga şi dreapta. Programul solicită ca date inițiale : lungimea grinzii, valoarea forței uniform distribuite, numărul total de forțe concentrate iar pentru fiecare forță concentrată, valoarea ei şi distanța față de limita stîngă a grinzii. Se determină valorile celor 2 momente de încastrare şi se afişează.

```
10 PRINT " MOMENT DE INCASTRARE PERFECTA "
20 PRINT " LUNGIMEA GRINZII IN M "
30 INPUT L
40 PRINT " FORTA UNIFORM DISTRIBUITA IN KGF/M "
50 INPUT P
60 T = P*L^2/12
70 R = T
80 PRINT " NR.-UL FORTELOR CONCENTRATE "
90 INPUT N
100 IF N = 0 THEN 280
110 DIM F(N),X(N)
120 PRINT " MARIMILE SI DISTANTELE FORTELOR "
125 PRINT " CONCENTRATE FATA DE STINGA GRINZII "
130 FOR J = 1 TO N
140 INPUT F(J)
150 INPUT X(J)
160 IF X(J) > L THEN 290
170 NEXT J
180 FOR J = 1 TO N
190 A1 = X(J)/L
200 A2 = 1-A1
210 V = A1*A2^2*F(J)*L
220 T = T+V
230 V = A1^2*A2*F(J)*L
240 R = R+V
250 NEXT J
260 PRINT "MST = ";T;" KGF.M "
270 PRINT "MDR = ";R;" KGF.M "
280 STOP
290 PRINT " ABSCISA GRESITA "
300 STOP
310 END
```

11.13. Optimizarea consumului de îngrăşăminte chimice în agricultură

Se porneşte de la funcţiile de producţie pentru terenuri irigate şi neirigate:

$$Y_{IRIG} = K_0a_0 + K_1a_1dN - K_2a_2dN^2$$

$$Y_{NEIRIG} = a_0 + a_1dN - a_2dN^2$$

unde:

dN este doza de îngrăşămînt la hectar

a_i sînt parametrii funcţiei de producţie

K_i sînt coeficienţi de multiplicare

```
10  PRINT " CONSUMUL DE INGRASAMINTE
20  PRINT " CHIMICE IN AGRONOMIE "
30  PRINT " DATI PARAMETRII AO,A1,A2
35  PRINT " A FUNCTIEI DE PRODUCTIE "
40  INPUT AO,A1,A2
50  PRINT " DATI COEFICIENTII KO,K1,K2 "
55  PRINT " DE MULTIPLICARE "
60  INPUT KO,K1,K2
70  N1 = A1/(2*ABS(A2))
80  N2 = K1*A1/(2*K2*ABS(A2))
90  N = N2-N1
100 INITP
110 PRINT AT(1,1);" FUNCTIA DE PRODUCTIE
120 PRINT AT(5,1);" Y(NEIRIG) = ";
125 PRINT AO;"+";A1;"*DN";A2;"*DN^2"
130 PRINT AT(7,1);" Y(IRIGAT) = ";
135 PRINT KO*AO;"+";K1*A1;"*DN";K2*A2;"*DN^2"
140 PRINT AT(9,1);" DOZA MAXIMA DE NEIRIG = ";
145 PRINT N1
150 PRINT AT(11,1);" DOZA MAXIMA IRIGAT = ";
155 PRINT N2
160 PRINT AT (13,1);" DIFERENTA = ";N
170 Y1 = AO+A1*N1-A2*(N1^2)
180 Y2 = KO*AO+K1*A1*N2-K2*A2*(N2^2)
190 Y = Y2-Y1
200 PRINT AT(15,1);" MAXIM TEHNIC NEIRIG = ";
205 PRINT Y1
210 PRINT AT(17,1);" MAXIM TEHNIC IRIGAT = ";
215 PRINT Y2
220 PRINT AT(19,1);" DIFERENTA = ";Y
230 X1 = Y1-AO
240 X2 = Y2-KO*AO
250 X = X2-X1
260 PRINT AT(22,1);" SPORUL MAXIM NEIRIG = ";
265 PRINT X1
270 PRINT AT(23,1);" SPORUL MAXIM IRIGAT = ";
275 PRINT X2
280 A$ = INKEY $
290 IF A$ = " " THEN 280
300 INITP
310 WINDOW -10,N2+30,-10,Y2+10
320 MOVE -10,0
330 DRAW N2+30,0
340 MOVE 0,10
350 DRAW 0,Y2+10
360 FOR I=25 TO 25*INT((N2+30)/25) STEP 25
370 MOVE I,-50
380 DRAW I,50
390 NEXT I
400 Q = INT((Y2+10)/1000)
405 FOR I=1000 TO 1000*INT(Y2*Q) STEP 1000
410 MOVE -50,I
420 DRAW 50,I
430 NEXT I
440 A = AO
450 B = A1
460 C = A2
480 FOR K = 1 TO 2
```

```
490 MOVE 0,0
500 FOR I=10 TO 10*INT((N2+30)/10) STEP 10
510 Y = A+B*I-C*I^2
520 DRAW I,Y
530 NEXT I
540 A = KO*AO
550 B = K1*A1
560 C = K2*A2
570 NEXT K
580 MOVE N1,0
590 DRAW N1,Y1
600 MOVE N2,0
610 DRAW N2,Y2
630 A$ = INKEY$
640 IF A$ = " " THEN 630
650 REM " CITIRE CONSTANTE "
660 READ A,B,C,D,E,F
670 INITP
680 DIM M(10,2)
690 MAT M = ZER
700 Z = 10
800 X = 0
810 WINDOW -10,300,-30,10000
820 MOVE -10,0
830 DRAW 300,0
840 MOVE 0,-10
850 DRAW 0,10000
860 FOR I = 1 TO 180 STEP 25
870 MOVE I, -50
880 DRAW I,50
890 NEXT I
900 FOR I=1000 TO 8000 STEP 1000
910 MOVE -50,I
920 DRAW 50,I
930 NEXT I
940 FOR Z = 10 TO 100 STEP 10
950 X = 0
960 GOSUB 2000
970 U = Z/10
980 MOVE X,Y
990 M(U,1) = X
1000 M(U,2) = Y
1010 FOR X = 10 TO 180 STEP 10
1020 GOSUB 2000
1030 DRAW X,Y
1040 IF Y < M(U,2) THEN 1070
1050 M(U,1) = X
1060 M(U,2) = Y
1070 NEXT X
1080 M(Z,10) = N
1090 NEXT Z
1100 MOVE M(1,1),M(1,2)
1110 FOR I = 2 TO 10
1120 DRAW M(I,1),M(I,2)
1130 NEXT I
1140 STOP
2000 Q = A*(Z^D)
2010 V = B*(Z^E)*X
2020 W = C*(Z^F)*(X^2)
2030 Y = Q+V-W
2040 RETURN
2050 DATA 36.1,17.11,0.796
2060 DATA 1.1087,0.11858,-0.60889
2070 END
```

Se determină doza maximă de îngrăşămînt pentru teren neirigat şi teren irigat. Se determină apoi maximul tehnic de producţie şi sporul maximi în cele două condiţii de lucru.

Se vizualizează graficul funcţiilor de producţie în condiţiile de irigare şi neirigare (ccoord. x=doze de îngrăşămînt în kg/ha ; coord. y=producţia medie în kg/ha). Particularizînd pentru funcţiile de producţie caracteristice culturilor de grîu, avem (vezi liniile 2050 şi 2060 din program):

$$Y=36{,}1\ NB^{1{,}1087}+17{,}11\ NB^{0{,}11858}\ dN-0{,}796\ NB^{-0{,}60989}\ dN^2$$

În baza acestei formule se trasează graficul producţiilor medii de grîu, în funcţie de notele de bonitare ale terenului agricol.

Modificarea liniilor 2050, 2060 permite calculul funcţiei de producţie pentru alte tipuri de culturi.

11.14. Calculul volumului rezervorului de compensaţie pentru reţeaua de apă potabilă

Variaţia consumului de apă pe parcursul unei zile se determină în baza debitului înregistrat la fiecare oră. Considerăm că alimentarea de la reţeaua de apă se face cu un debit mediu constant ; în acest caz, rezervorul de compensaţie va trebui să acumuleze apa necesară consumului în orele de vîrf.

Programul solicită mai întîi introducerea valorilor celor 24 de debite orare. Se determină apoi debitul mediu, pe baza formulei :

$$Dm = \frac{1}{24} \sum_{i=1}^{24} Di$$

Rezultă volumul rezervorului de compensaţie :

$$Vr = \sum_{i=1}^{24} (Di-Dm)*((SGN\ (Di-Dm)+1)/2)$$

Se afişează consumul mediu şi volumul rezervorului de compensaţie.

```
10 PRINT " REZERVORUL DE COMPENSATIE "
20 DIM D(24)
30 PRINT " DATI CONSUMURILE PE ORE "
40 MAT INPUT D
50 X = O
60 FOR I = 1 TO 24
70 X = X+D(I)
80 NEXT I
90 X = X/24
100 V = O
110 FOR I = 1 TO 24
120 IF (D(I)-X) <= O THEN 140
130 V = V+(D(I)-X)
140 NEXT I
150 PRINT " CONSUM MEDIU ";X
160 PRINT " VOLUMUL REZERVORULUI ":V
170 STOP
180 END
```

11.15. Studiul unui filtru „trece-jos"

Fiind dat filtrul RC „trece-jos" din figura 11.3, ecuaţia va fi

$$U_1(t) = U_2(t) + RC\frac{dU_2}{dt}.$$

Ecuaţia poate fi aproximată cu :

$$U_1(n) = U_2(n) + \frac{RC}{\Delta t}[U_2(n) - U_2(n-1)]$$

unde $RC/\Delta t = T$ este constanta de timp normată a filtrului. Rezultă relaţia de recurenţă :

$$U_2(n) = \frac{U_1(n)}{1+T} + \frac{T}{1+T}U_2(n-1)$$

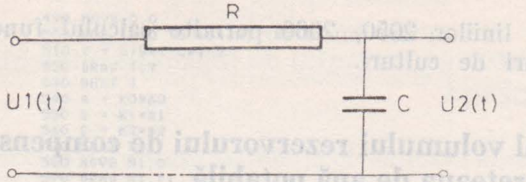


Fig. 11.3. Filtru RC „trece-jos".

Forma semnalului de intrare este definită în cadrul programului în liniile 150—180, obţinîndu-se la ieşire un semnal ca în figura 11.4. Programul solicită introducerea constantei de timp normate şi a valorii iniţiale a semnalului de ieşire U_2. Se trasează graficul suprapus a celor două semnale, obţinîndu-se diferite forme ale semnalului U_2 în funcţie de constanta de timp T; regimul tranzitoriu poate fi observat cu uşurinţă pe grafic.

```
10  PRINT " STUDIUL FILTRULUI "
20  PRINT " DATI CARACTERISTICA T "
30  INPUT T
40  PRINT " DATI VALOAREA INITIALA U2 "
50  INPUT U2
60  INITP
70  WINDOW -10,50,-1,2
80  MOVE -10,0
90  DRAW 50,0
100 MOVE 0,-1
110 DRAW 0,2
120 U1 = 1
130 MOVE 0,U1
140 FOR X = 1 TO 50
150 IF SGN(X-INT(X/10)*10-5 <= 0 THEN 180
160 U1 = 0
170 GOTO 190
180 U1 = 1
190 DRAW X,U1
200 MOVE X-1,U2
210 U2 = U1/(1+T)+(T/(1+T))*U2
220 DRAW X,U2
230 MOVE X,U1
240 NEXT X
250 STOP
260 END
```

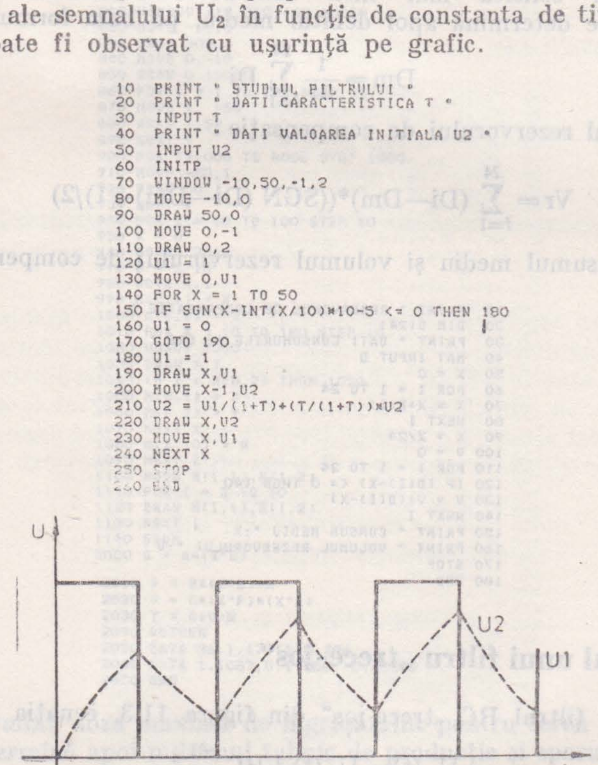


Fig. 11.4. Semnalul la ieşirea filtrului „trece-jos".

11.16. Calculul salinităţii unui canal de ecluză

Programele rezolvă determinarea salinităţii apei într-un canal de ecluză ce leagă două ape cu salinităţi diferite. Se introduc datele iniţiale şi în diferite ipoteze de lucru se obţine numărul de ecluzări necesare.

```
5 PRINT " P R G R M 1 "
10 REM."CALCULUL SALINITATII CANALULUI"
15 REM "IPOTEZA 1 = DIFERENTA V = 0"
20 REM "CITIRE DATE INITIALE"
30 READ C1,C2,C3,D,H1,H2,A,V,E
35 REM "CALCUL COEFICIENTI"
40 D1=D/V
50 D2=D/(A*H1)
60 D3=D/(A*(H1+H2))
65 REM "CONSTANTE"
70 X=1-D3
80 Y=1-D2
90 Z=1/(X*Y)
95 REM "SALINITATE ADMISA S A S = S"
100 A=C1*(1+Z)+C2/X+C3*(1-D3*Y)*Z
110 B=(C3*C2-C1*(D2*C2+D3*Y*C3))*Z
120 K=A/2
130 D=K*K-B
220 S=K-SQR(D)
225 REM "SALINITATE INITIALA S A S = M"
230 M=D2*C2+S*X*Y+D3*Y*C3
235 REM "ADINCIMEA REMANENTA"
240 H=X*(S-C1)/(C2-C1)
245 REM "VARIATIA SALINITATII"
250 N=0
260 I=C1
270 W=C1
280 N=N+1
290 V=H*(I-W)/X+W
300 I=D2*C2-D3*Y*(V-W)+V*Y
310 W=D1*V+(1-D1)*W
315 REM "VERIFICARE LIMITE""
320 IF I>=M-E THEN 360
330 IF V>=S-E THEN 360
340 IF W>=C3-E THEN 360
350 GOTO280
360 PRINT " NUMARUL DE ECLUZARI ";N
370 STOP
375 REM "DATE INITIALE"
380 DATA 0.15,17,0.5
390 DATA 21000,7,9
400 DATA 8000,36.0E6
410 DATA 0.01
3660 END
```

```
5 PRINT " P R G R M 2 "
10 REM "CALCULUL SALINITATII CANALULUI"
15 REM "IPOTEZA 2 = DIFERENTA V > 0"
20 REM "CITIRE DATE INITIALE"
30 READ C1,C2,C3,D,H1,H2,A,V1,V2,E
35 REM "CALCUL COEFICIENTI"
40 D1=D/V1
50 D2=D/(A*H1)
60 D3=D/(A*(H1+H2))
65 D4=V2/V1
70 D5=(1-D4)*(1-D1)
72 REM "SALANITATE ADMISA S A S = S"
75 S=((1-D5)*C3-D4*C1)/(D1*(1-D4))
80 S1=D2*C2+S*(1-D2)+(1-D3)+D3*(1-D2)*C3
90 R=((1-D5)*(1-D3)*(C3-C1))/(D1*(1-D4)*(C2-C1))
95 REM "VARIATIA SALINITATII"
100 N=0
110 I=C2
120 W1=C1
125 W2=C1
130 N=N+1
140 V=R*(I-W2)/(1-D3)+W2
150 I=D2*C2-D3*(1-D2)*(V-W2)+V*(1-D2)
160 W2=(1-D4)*D1*V+D5*W1+D4*C1
170 W1=D1*V+(1-D1)*W2
175 REM " VERIFICARE LIMITE"
180 IF N<100 THEN 130
185 IF I>=S1-E THEN 220
190 IF V>=S-E THEN 220
200 IF W2>=C3-E THEN 220
205 GOTO 130
210 REM "AJUNS LIMITA"
220 PRINT N,I,V,W
230 STOP
290 REM "DATE INITIALE"
300 DATA 0.15,17,0.5
310 DATA 21000,7,9
320 DATA 8000,36.0E6,12.0E4
330 DATA 0.01
340 END
```

```
5 PRINT " P R G R M 3 "
10 REM "DISTRIBUTIA SALINITATII IN CANAL"
15 REM "IPOTEZA 1 = DIFERENTA V = 0"
20 REM "CITIRE DATE INITIALE"
30 READ C1,C2,C3,D,H1,H2,A,V,L,T,V1,E,N1,N2
35 REM "CALCUL COEFICIENTI"
40 D1=D/V
45 D2=D/(A*H1)
50 D3=D/(A*(H1+H2))
55 D4=L/V1
60 D5=D4/T
65 DIM K(250),L(250)
70 Y=1-D2
80 X=1-D3
90 Z=1/(X*Y)
95 REM "SALINITATEA ADMISA IN S A S = S"
100 A=C1*(1+Z)+C2/X+C3*(1-D3*Y)*Z
110 B=(C3*C2-C1*(D2*C2+D3*Y*C3))*Z
120 U=A/2
130 M=U*U-B
200 S=U-SQR(M)
220 REM "SALINITATEA INITIALA S A S = M"
230 M=D2*C2+S*X*Y+D3*Y*C3
235 REM "ADINCIMEA REMANENTA = H"
240 H=X*(S-C1)/(C2-C1)
245 REM "VARIATIA SALINITATII"
250 J=1
260 N=0
270 MAT K=ZER
280 MAT L=ZER
290 I=C2
300 W=C1
310 N=N+1
320 V=H*(I-W)/X+W
330 I=D2*C2-D3*Y*(V-W)+V*Y
340 W=D1*V+(1-D1)*W
350 IF N<N1 THEN 310
360 K(J)=V
370 L(J)=W
380 J=J+1
390 IF J<=D5 THEN 310
400 W=L(1)
410 U=W
420 FOR J=2 TO D5
430 M=L(J-1)+(D/(V*(D5-J+1)))*(K(J)-W)
440 IF M<U THEN 455
450 U=W
455 W=L(J)
460 NEXT J
465 REM "TIPARIRE VARIATIA MAXIMA"
470 PRINT U
475 REM "VERIFICARE LIMITA"
480 IF U>C3-E THEN 510
490 N1=N1+N2
500 GOTO 250
510 STOP
590 REM "DATE INITIALE"
600 DATA 0,15,17,0.5
610 DATA 21000,7,9
620 DATA 8000,36.0E6
630 DATA 57600,3600,1
640 DATA 0.01,400,100
650 END
```

11.17. Calculul hidraulic al ecluzelor

Programele determină caracteristicile hidraulice principale ale unei ecluze, funcţie de condiţiile de evacuare a apei.

```
5 PRINT " P P G R M S "
10 REM "CALCULUL HIDRAULIC AL ECLUZELOR"
15 REM "EVACUARE IN BIEF. AVAL"
20 REM "CITIRE DATE INITIALE"
30 READ A3,H,A1,C1,R1,T1,A2,C2,R2,T2,V,E
35 D=H
40 I=0
45 T=0
50 T3=0
55 T5=0
60 J=20
65 L=0
70 B=SQR(2*9.81)
75 U2=C2
80 N=(A1/A3)*B*J
85 O=(A2/A3)*B*J
90 Q1=0
95 P=0
100 Z=0
130 I=I+1
140 T=T+J
150 IF T>R1 THEN 210
160 M=I*J*C1/R1
170 U1=(L+M)/2
180 L=M
190 GOSUB 400
200 GOTO 130
210 IF T3<>0 THEN 250
220 U1=C1
230 GOSUB 400
240 GOTO 130
250 L=C2
260 T4=T3+T2
270 M=(T4-I*J)*C2/T4
280 U2=(L+M)/2
285 I=I+1
290 T=T+J
295 IF T>T4 THEN 320
300 L=M
305 GOSUB 400
310 GOTO 270
320 U=D
330 R=SQR(U)-C1*A1*B*(I*J-T4)/(2*A3)
340 Q3=C1*A1*B*SQR(R)
345 D=D-R
350 PRINT I,Q3,D
360 I=I+1
370 T=T+J
380 IF D>E THEN 330
390 STOP
395 REM "SUBPROGRAM CALCUL ITERATIV SARCINA"
400 R=N*U1*SQR(D-P/2)-O*U2*SQR(H-(D-P/2))
410 IF ABS(R-P)<=E THEN 440
420 P=R
430 GOTO 400
440 P=R
445 REM "DIFERENTA DE SARCINA"
450 D=D-R
455 REM "DEBIT AFLUENT"
460 Q2=U2*A2*B*SQR(H-D)
465 REM "DEBIT EFLUENT"
470 Q3=U1*A1*B*SQR(D)
475 REM "VOLUMUL EVACUAT"
480 X=(Q1+Q2)/2
490 X=X*J
495 REM "SARCINA ECLUZA"
500 Z=Z+X
510 PRINT I,Z,Q2,Q3,C2,D
520 Q1=Q2
530 IF Z<= 0.9*V THEN 550
540 T3=T
550 RETURN
555 REM "DATE INITIALA"
600 DATA 2000,1.75
610 DATA 6.41,0.5,180,120
620 DATA 8.75,0.65,360,120
630 DATA 11000,0.01
640 END
```

```
5 PRINT " P R G R M 6 "
10 REM "CALCUL HIDRAULIC AL ECLUZEI"
15 REM "EVACUARE IN BIEF AVAL"
20 REM "CONDITIE STATIONARE 3 A 8"
25 REM "CITIRE DATE INITIALE"
30 READ A3,H,A1,C1,R1,T1,A2,C2,V,E
40 D=H
45. I=0
50 T=0
55 T3=0
60 T5=0
65 J=20
70 L=0
75 B=SQR(2*9.81)
80 U2=C2
85 N=(A1/A3)*B*J
90 O=(A2/A3)*B*J
95 Q1=0
100 P=0
105 Z=0
130 I=I+1
140 T=T+J
150 IF T>R1 THEN 210
160 M=I*J*C1/R1
170 U1=(L+M)/2
180 L=M
190 GOSUB 400
200 GOTO130
210 U1=C1
215 U=D
220 C=Q3-Q2
230 T3=T
240 R=N*C1*SQR(U-P/2)-O*C2*SQR(H-(U-P/2))
250 D=D-R
260 Q3=C1*A1*B*SQR(D)
270 K=(Q3-C)/(A2*B*SQR(H-D))
280 C2=K
290 P=R
300 IF ABS(R-P)>E THEN 240
310 GOSUB 460
320 I=I+1
330 T=T+J
340 IF C2>E THEN 240
350 U=D
355 R=SQR(U)-C1*A1*B*(I*J-T3)/(2*A3)
360 Q3=C1*A1*B*SQR(R)
365 D=D-R
370 PRINT I,Q3,D
375 I=I+1
380 T=T+J
385 IF D>E THEN 355
390 STOP
395 REM "SUBPROGRAM CALCUL ITERATIV SARCINA"
400 R=N*U1*SQR(D-P/2)-O*U2*SQR(H-(D-P/2))
410 IF ABS(R-P)<=E THEN 440
420 P=R
430 GOTO400
440 P=R
445 REM "DIFERENTA DE SARCINA"
450 D=D-R
455 REM "DEBIT AFLUENT"
460 Q2=U2*A2*B*SQR(H-D)
465 REM "DEBIT EFLUENT"
470 Q3=U1*A1*B*SQR(D)
475 REM "VOLUMUL EVACUAT"
480 X=(Q1+Q2)/2
490 X=X*J
495 REM "SARCINA ECLUZA"
500 Z=Z+X
510 PRINT I,Z,Q2,Q3,C2,D
520 Q1=Q2
530 RETURN
590 REM "DATE INITIALE"
600 DATA 2000,1.75
610 DATA 6.41,0.5,180,120
620 DATA 8.75,0.65
630 DATA 11000,0.01
640 END
```

```
5 PRINT " P R G R M 7 "
10 REM "CALCUL HIDRAULIC AL ECLUZEI"
15 REM "EVACUARE IN BAZIN SUBTERAN"
20 REM "CITIRE DATE INITIALE"
30 READ A3,H1,H2,Z,H,S,C,A1,C1,R1,T1,A2,C2,R2,T2,V,E
35 REM "INITIALIZARI"
40 D1=H1
41 I=0
45 T=0
50 T3=0
55 T5=0
60 J=20
65 L=0
70 B=SQR(2*9.81)
75 X=0
80 N=(A1/A3)*B*J
85 O=(A2/A3)*B*J
90 Q1=0
95 P=0
100 H1=H1+H2+Z
105 F=0
110 D2=H2
115 Z=0
130 I=I+1
135 T=T+J
140 IF T>R1 THEN 200
145 M=I*J*C1/R1
150 U1=(L+M)/2
155 L=M
160 IF T>R2 THEN 190
165 Y=I*J*C2/R2
170 U2=(X+Y)/2
175 X=Y
180 GOSUB 400
185 GOTO130
190 U2=C2
195 GOTO180
200 IF T3<>0 THEN 230
205 U1=C1
210 U2=C2
215 GOSUB 400
220 GOTO130
230 X=C2
235 T4=T3+T2
240 Y=(T4-I*J)*C2/T4
245 U2=(X+Y)/2
250 I=I+1
255 T=T+J
260 IF T>T4 THEN 280
265 X=Y
270 GOSUB 400
275 GOTO240
280 U=D1
290 R=SQR(U)-C1*A1*B*(I*J-T4)/(2*A3)
300 Q3=C1*A1*B*SQR(R)
310 D1=D1-R
320 PRINT I,Q3,D1
330 I=I+1
340 T=T+J
350 IF D1>E THEN 290
360 STOP
390 REM "SUBPROGRAM CALCUL ITERATIV SARCINA"
400 R=N*U1*SQR(D1-P/2)-U*U2*SQR(H1-(D1-P/2)-(D2-F/2))
405 W=(H+C*(2*D2+F))*(S+C*(2*D2+F))
410 G=U2*A2*B*SQR(H1-(D1-P/2)-(D2-F/2))*J/W
420 IF ABS(P-R)<=E THEN 460
430 P=R
440 F=G
450 GOTO 400
460 P=R
470 F=G
475 REM "DIFERENTE DE SARCINA"
480 D1=D1-R
490 D2=D2-G
492 REM "DEBIT EFLUENT"
495 Q3= U1*A1*B*SQR(D1)
497 REM "DEBIT AFLUENT"
500 Q2=U2*A2*B*SQR(H1-D1-D2)
505 REM "VOLUM EVACUAT"
510 K=(Q1+Q2)/2
520 K=K*J
525 REM "SARCINA ECLUZA"
530 Z=Z+K
540 PRINT I,Z,Q2,Q3,C2,D1
550 Q1=Q2
560 IF Z<=0.9*V THEN 580
570 T3=T
580 RETURN
600 DATA 2000,1,75
610 DATA 7,1.5
620 DATA 7,9,10
630 DATA 6.41,0.5,180,120
640 DATA 8.75,0.65,360,120
650 DATA 11000,0.01
660 END
4922
```

Capitolul 12.

Microcalculatorul personal aMIC în procesul de învăţămînt *

Încercările mai vechi de utilizare a calculatoarelor în procesul de învăţămînt s-au bazat pe folosirea terminalelor de tip display, cuplate la un calculator central. Deşi utile, sistemele nu s-au extins în învăţămîntul liceal, chiar în ţările avansate industrial, datorită costurilor lor ridicate.

Apariţia calculatoarelor personale permite abordarea acestei probleme de pe noi poziţii, deoarece aceste calculatoare pot fi folosite de către elevi, atît în şcoală, cît şi la domiciliu. Facilităţile hardware şi software de care dispun calculatoarele personale permit cuplarea lor la un calculator central, dotat cu o bază de date (cunoştinţe). În acest context asistarea procesului de învăţămînt de către calculator capătă noi dimensiuni.

12.1. Modalităţi de integrare a calculatorului în procesul de predare-învăţare, locul şi rolul acestuia în asistarea procesului de învăţămînt

Asistarea procesului de învăţămînt cu calculatorul. Această problemă este de dată recentă, dar de mare perspectivă şi cu consecinţe pozitive pentru optimizarea predării-învăţării, pentru integrarea învăţămîntului cu cercetarea, cu producţia, cu viaţa.

Este cunoscut faptul că în ultimele două decenii, colaborarea dintre informaticieni, constructori de calculatoare şi specialişti din domenii ale instrucţiei şi educaţiei a permis iniţierea unor programe concrete privind folosirea calculatoarelor în procesul de învăţămînt. Ca urmare, au fost concepute, realizate şi perfecţionate variate limbaje de programare, cu ajutorul cărora să se poată edifica autoinstruirea automată, să fie prezentate la clasă lecţii, ori secvenţe ale acestora, în sprijinul optimizării procesului de predare-învăţare. În consecinţă, psihopedagogiei i-a revenit sarcina edificării unei metodologii care să asigure eficienţa asistării procesului de învăţămînt cu calculatorul.

Conceptul de asistare cu calculatorul a procesului de învăţămînt include atît predarea cu calculatorul a unei lecţii de comunicare a noilor cunoştinţe, de aplicare, de consolidare ori de sistematizare a acestora, cît şi verificarea

---

* **Cu orientare spre învăţămîntul mediu.**

automată a unei lecţii sau a unui grup de lecţii, a unei anumite discipline şcolare la finele unui trimestru sau an şcolar, a unei anumite programe şcolare în cadrul examenelor de absolvire sau de admitere. Tot în cadrul „asistării" este inclusă şi prezentarea sau verificarea cu calculatorul a unor secvenţe ale lecţiilor desfăşurate după metodologia clasică.

Calculatorul devine un auxiliar al procesului de învăţămînt.

Tehnica instruirii cu calculatorul. Instruirea cu ajutorul calculatorului se realizează fie în cadrul aşa-ziselor „clase automatizate", în care lecţia se derulează secvenţă cu secvenţă pe ecranul de la pupitrul fiecărui elev, fie în cadrul laboratoarelor de matematică — mai ales, pentru prezentarea de la catedră, prin intermediul ecranului, a unor secvenţe ale lecţiei clasice. O clasă automatizată dispune de un calculator şi mai multe terminale (în jur de 3Ø) care recepţionează mesajele pe ecranul individual.

Predarea lecţiei prin intermediul ecranului se realizează printr- o succesiune de imagini, numite „imagini-ecran", care afişează lecţia, derularea lor fiind dirijată de către profesor — în concordanţă cu normele psihopedagogice ale procesului de instruire la vîrsta şi în cadrul obiectului predat.

Instruirea cu ajutorul calculatorului se poate însă realiza şi în afara schemei clasice a procesului de învăţămînt. Elevul, studentul, sau orice altă persoană interesată poate beneficia, într-un anumit cadru, de posibilitatea autoinstruirii automatizate. Aceasta se realizează fără profesor, activitatea sa fiind însă materializată în lecţiile afişate pe ecran. Dialogul între maşină şi cel care învaţă singur se realizează cu ajutorul unei console legată de calculator, aceasta fiind construită dintr-o claviatură alfanumerică şi din ecranul semnalat mai sus.

Lecţia, sau grupul de lecţii, sînt stocate în memoria calculatorului şi cu ajutorul software-ului sînt afişate treptat, imagine cu imagine, pe ecranul consolei, furnizînd astfel materialul de învăţat.

În cadrul acestei instruiri elevul îşi poate regla singur ritmul de afişare a imaginilor pe ecran ; el îşi poate întrerupe pregătirea în orice moment al lecţiei, reluînd-o de la secvenţa la care s-a oprit, pe baza „recunoaşterii" sale de către calculator. Această posibilitate de „recunoaştere" permite folosirea în timp a aceleiaşi lecţii de către mai mulţi elevi.

În lecţiile predate cu ajutorul calculatorului sînt prevăzute şi exerciţii de control, răspunsurile putînd fi adeseori selectate dintr-o listă de variante afişată pe terminal. Metoda folosită în acest caz este, oarecum, similară *învăţămîntului programat*. În cazul că lecţia urmăreşte testarea cunoştinţelor, atunci fiecărei imagini-ecran i se asociază una sau mai multe întrebări. Răspunsurile la întrebări sînt cotate cu un anumit punctaj, acesta depinzînd de calitatea răspunsului şi de timpul în care a fost dat.

În dialogul dintre maşină şi cel care învaţă se foloseşte şi funcţia de „help" a calculatorului. Prin aceasta sînt puse la dispoziţia utilizatorului, sub formă de imagini-ecran, comentariile şi explicaţiile necesare integrării şi însuşirii corecte a unor noţiuni — fără de care lecţia nu poate continua. Evidenţiem, cu acest prilej, „posibilitatea" maşinii de a detecta imediat eroarea în însuşirea unei secvenţe a lecţiei şi de a facilita remedierea acesteia prin comenta-

riul făcut şi indicarea informaţiilor suplimentare necesare continuării instruirii. Cel care învaţă poate trece la secvenţa următoare a lecţiei numai dacă a răspuns corect la întrebările sau exerciţiile propuse, interacţiunea dintre utilizator şi sistem realizîndu-se permanent pe bază de dialog.

Calculatorul didactic poate fi înzestrat şi cu „posibilitatea" de a memoriza performanţele celui ce învaţă : timpul necesar pentru răspuns, nota obţinută, întrebările la care frecvenţa răspunsurilor slabe este mare etc. În acest mod sînt furnizate fie elevului, fie profesorului, o serie de date pedagogice individuale, sau caracteristice unei anumite populaţii şcolare, care sînt necesare optimizării acestor programe şi lecţiilor recapitulative sau de control curent ce se vor organiza.

Consecinţe pedagogice ale asistării lecţiilor cu calculatorul. Folosirea calculatorului în şcoală oferă învăţării lecţiei noi posibilităţi de dezvoltare şi de evaluare. Aşa, spre exemplu, prin intermediul consolei pot fi simulate pe ecran procese şi fenomene în evoluţia lor, unele experienţe greu accesibile laboratoarelor şcolare fie datorită costului lor prea ridicat, fie pericolelor existente, protecţiei muncii şi a mediului înconjurător, fie depărtării acestora în timp sau în spaţiu etc.

Ca o nouă componentă a tehnologiei învăţămîntului, folosirea calculatoarelor în munca instructiv-educativă va lărgi aria de posibilităţi şi de funcţionalitate a laboratoarelor şcolare, în beneficiul tuturor disciplinelor de învăţămînt care folosesc tehnică de calcul sau de reprezentare.

Cu ajutorul calculatorului poate fi optimizat randamentul predării prin prezentarea cu ajutorul ecranului a unei largi varietăţi de exemple sau de modele asociate unor secvenţe ale lecţiei. Toate acestea concură la adîncirea sau lărgirea orizontului noţiunilor predate, adeseori extrapolîndu-le dincolo de obiectul predat : în tehnică, în economie, în ştiinţă, în practică etc. În acest mod s-ar putea vorbi despre o „dilatare" a sferei aplicative a noţiunilor predate, precum şi de o „comprimare" a timpului necesar însuşirii şi aplicării creatoare a noţiunilor ştiinţei. Acest aspect poate conduce la stimularea inventivităţii şi aplicativităţii, a spirirtului participativ şi anticipativ al celui ce învaţă.

Folosirea calculatoarelor ca mijloace de învăţămînt va avea, desigur, consecinţe importante asupra formării intelectuale a tinerelor generaţii în spiritul autoeducaţiei — şi, prin aceasta, al educaţiei permanente. În această activitate formativă vor fi stimulate toate componentele gîndirii logice în sens larg : gîndirea logico-deductivă, inductivă, analogică, . . . cu accent pe gîndirea euristică. Va fi astfel stimulată mai bine şi problema orientării profesionale a tineretului, capacitatea acestuia de reciclare rapidă pentru sectoare de muncă „înrudite" profesional. Fireşte, pentru atingerea acestui ţel, considerăm că nici disciplinele tehnologice şi nici atelierele-şcoală, prin şi în care elevii trebuie să înveţe să producă la nivelul tehnicii contemporane, nu trebuie să rămînă în afa a folosirii calculatorului şi a tehnicilor de calcul ale producţiei moderne. Prin aceasta s-ar putea realiza mai bine şi componenta tehnică a educaţiei multilaterale a personalităţii umane.

Desigur, progresele realizate astăzi pe linia dezvoltării şi a ieftinirii circuitelor integrate, între care microprocesoarele ocupă un loc important, vor per-

mite construirea pe scară tot mai largă a unor echipamente aşa-zise de "uz personal", numite şi calculatoare personale (individuale). Acestea pot fi folosite şi în procesul de învăţămînt, intrînd — pentru început — în zestrea laboratoarelor de matematică (de ex.), în calitate de auxiliar al predării, al verificării cunoştinţelor, ..., al învăţării.

Dar pentru integrarea calculatorului în familia mijloacelor de învăţămînt, profesorului i se cere să stăpînească, în afara specialităţii sale, un volum apreciabil de cunoştinţe din domeniul informaticii, un limbaj de programare a lecţiilor sale, cum şi o mare varietate de tehnici pentru realizarea desenelor şi a exerciţiilor din lecţiile sale curente, de sinteză sau de control etc. Se întrezăreşte de aci necesitatea acestor cunoştinţe, cum şi a unor deprinderi tehnice, în pregătirea generală a întregului personal didactic din învăţămîntul secundar. Acestea se pot obţine pe baza unor programe speciale, atît în pregătirea universitară a cadrelor cît şi în perfecţionarea postuniversitată. Mai mult, pentru folosirea calculatorului va fi necesară o pregătire generală a întregului tineret — în cadrul noilor programe ale învăţămîntului liceal. Fireşte, este necesar ca — în perspectivă — să dispunem de limbaje de programare foarte simplificate, ameliorînd astfel eforturile utilizatorilor pentru însuşirea lor.

Trebuie recunoscut că nici pentru elev nu este chiar atît de simplă şi de uşoară pregătirea individuală cu ajutorul calculatorului. Şi acestuia îi este mai uşor dacă poate primi cunoştinţele în limba sa proprie, dacă poate "conversa" cu maşina într-un limbaj natural — nu artificial. În plus, maşina care serveşte autoeducaţiei automatizate nu poate dispune de prea multe mijloace pedagogice, în afara celor ce decurg din evidenţierea reuşitei la învăţătură, sau a semnalării insuccesului şi trimiterea la întrebări şi materiale ajutătoare.

Este de apreciat faptul că, însuşindu-şi cunoştinţe în faţa consolei, elevul învaţă singur şi sigur, adeseori în ritm optim propriu, fără emoţii şi fără perturbări ale comportamentului de către diverşii factori legaţi de mediul său înconjurător. El este scutit de emoţiile care apar atunci cînd se vede "urmărit" de către profesor, sau de către colegii săi de clasă (deşi, aceste emoţii pot avea frecvent şi un rol pozitiv asupra personalităţii sale).

De asemenea, în faţa consolei elevul primeşte obiectiv şi cu optimism „nota" acordată răspunsurilor şi deci pregătirii sale ; învaţă de aici să-şi aprecieze calitatea şi durata pregătirii, performanţele atinse şi, mai ales, învaţă să-şi mobilizeze resursele de energie şi de voinţă pentru o nouă calitate a muncii sale.

După cum se vede, ştiinţei învăţării i se circumscrie un nou domeniu în care cercetarea pedagogică interdisciplinară va fi solicitată foarte mult. Absenţa acesteia poate frîna accesul calculatoarelor în domeniul învăţămîntului.

Încă de pe acum au apărut o serie de întrebări în legătură cu aşa-zisa *"înstrăinare"* a individului ca urmare a unui învăţămînt automatizat.

Noi considerăm că un învăţămînt complet automatizat nu poate exista, problema "înstrăinării" pierzîndu-şi astfel suportul pe baza căruia a fost lansată. Practica va fi, desigur, în favoarea folosirii tehnologiilor celor mai avansate şi în domeniul învăţămîntului, atît dezvoltarea tehnicii cît şi a ştiinţei educaţiei fiind strîns legate de dezvoltarea întregii societăţi. Or, societatea contribuie

nu numai la umanizarea tot mai înaltă a personalităţii umane, ci şi a tehnicii, a matematicii chiar, a mijloacelor de învăţămînt şi educaţie. Mai mult, există şi o întoarcere ''reflexivă : tehnica, matematica, ..., au la rîndul lor un rol tot mai mare în umanizarea omului.

12.2. Modele de lecţii sau secvenţe ale acestora pentru instruirea asistată de calculator (IAC) a matematicii în liceu

Tema. Rezolvarea sistemelor de două ecuaţii de gradul I cu două necunoscute (lecţie de recapitulare ţinută în laboratorul de matematică la începutul clasei a IX-a)

În această lecţie se urmăreşte ca elevii să alcătuiască corect programul de rezolvare a unui sistem de două ecuaţii de gradul I cu două necunoscute şi schema logică a rezolvării sistemului ; de asemenea, ei trebuie să se convingă de necesitatea folosirii calculatorului la rezolvarea unor sisteme care necesită un volum mare de calcul.

1° Programul de rezolvare a sistemului.

Pe baza recapitulării cu clasa, a principalelor noţiuni şi metode privind rezolvarea sistemelor de două ecuaţii de gradul I cu două necunoscute, se stabileşte programul de rezolvare a sistemului

$$\begin{cases} A\cdot X+B\cdot Y=C \\ D\cdot X+E\cdot Y=F. \end{cases}$$

Astfel se notează : $M=A\cdot E-B\cdot D$, $M1=C\cdot E-F\cdot B$, $M2=A\cdot F-C\cdot D$ şi se face discuţia de mai jos :

dacă $M=\emptyset$ şi $M1\neq\emptyset$, atunci sistemul este incompatibil ;

dacă $M=\emptyset$ şi $M1=\emptyset$, atunci sistemul este compatibil nedeterminat şi are o infinitate de soluţii ;

dacă $M\neq\emptyset$, atunci sistemul este compatibil determinat şi are soluţiile

$$X=\frac{M1}{M},\ Y=\frac{M2}{M}.$$

2°. Schema logică

În perspectivă, noile programe de matematică vor beneficia — probabil de noţiunile privind schema logică, încă din clasele VII—VIII. În acest caz, ţinînd seama de program, se va alcătui cu clasa schema logică a rezolvării sistemului.

Observaţie. În cazul că programa nu prevede noţiunea de schemă logică, se va trece la rezolvarea unor sisteme de ecuaţii, verificîndu-se soluţiile cu calculatorul. (fig. 12.1)

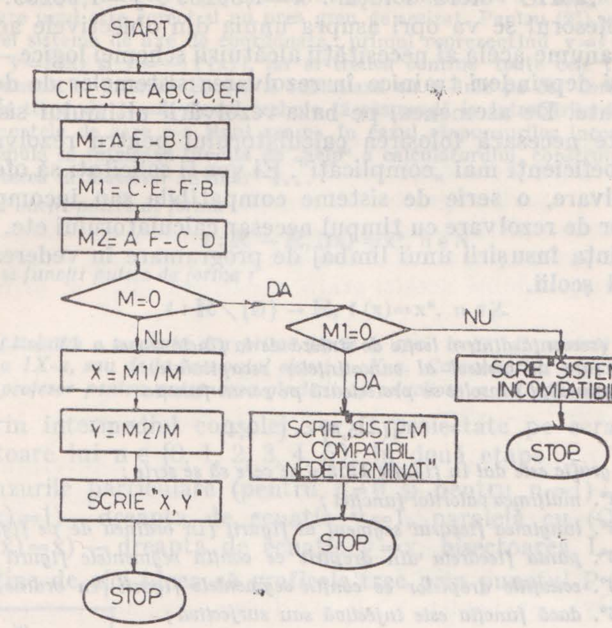


Fig. 12.1. Organigrama rezolvării unui sistem de două ecuații cu două necunoscute.

3°. Programul în BASIC

```
1 INITP
5 PRINT "REZOLVAREA UNUI SISTEM DE"
10 PRINT "DOUA ECUAȚII CU DOUA"
15 PRINT "NECUNOSCUTE"
20 PRINT "A*X+B*Y=C ; D*X+E*Y=F"
30 INPUT A,B,C,D,E,F
40 M=A*E—B*D
50 M1=C*E—F*B
60 M2=A*F—C*D
70 IF M=Ø THEN 13Ø
80 X=M1/M
90 Y=M2/M
100 PRINT "SOLUȚIILE SISTEMULUI"
110 PRINT "X=",X ; "Y=",Y
120 GO TO 18Ø
130 IF M1< >Ø THEN 16Ø
140 PRINT "SISTEM COMPAT. NEDETER-
     MINAT"
150 GO TO 18Ø
160 PRINT "SISTEM INCOMPATIBIL"
180 END.
```

4°. Aplicație pe calculator

Se cere rezolvarea sistemului :

$$\begin{cases} 5{,}625x+375y=721{,}851 \\ 105x+3{,}25y=163{,}9348 \end{cases}$$

Practic, o parte din elevi vor da soluții greșite în rezolvarea acestui sistem. Soluția este : $x=1{,}5Ø24$; $y=1{,}9Ø24$

Se trece apoi la rezolvarea cu calculatorul a sistemului.

Calculatorul „aMIC" oferă soluția : x=1,5Ø239 ; y=1,9Ø239.

În final, profesorul se va opri asupra unuia din obiectivele acestei lecții de recapitulare și anume acela al necesității alcătuirii schemei logice — ca dovadă a unor priceperi și deprinderi trainice în rezolvarea sistemelor de două ecuații cu două necunoscute. De asemenea, pe baza rezolvării ultimului sistem, elevii se conving că este necesară folosirea calculatorului pentru rezolvarea unora din sistemele cu coeficienți mai „complicați". Ei vor fi solicitați să ofere calculatorului, spre rezolvare, o serie de sisteme compatibile sau incompatibile, să compare timpul lor de rezolvare cu timpul necesar calculatorului etc. Pe această bază apare și dorința însușirii unui limbaj de programare în vederea accesului lor la calculatorul școlii.

Tema. Funcția (secvență dintr-o lecție de sinteză de la finele clasei a IX-a — ea poate servi ca o lucrare de control al cunoștințelor recapitulate). Prin intermediul consolei se proiectează pe ecran funcția

$$f : (-5,6] \to (-4,4]$$

al cărei grafic este dat în figura 12.2 și se cere să se scrie ;

1°. mulțimea valorilor funcției ;

2°. lungimea fiecărui segment al figurii (în ordinea de pe figură) ;

3°. panta fiecăreia din dreptele ce conțin segmentele figurii (în ordine) ;

4°. ecuațiile dreptelor ce conțin segmentele figurii (în ordine) ;

5°. dacă funcția este injectivă sau surjectivă ;

6°. mulțimea valorilor argumentelor pentru care se anulează funcția ;

7°. mulțimea valorilor argumentelor pentru care $f(x) \geq 0$ (respectiv $f(x) < 0$) ;

8°. mulțimea pentru care funcția este strict crescătoare (respectiv strict descrescătoare) ;

9°. reprezentarea în același sistem a funcțiilor date de $y = {}' f_a x_b{}'$ și $y = -{}' f_a x_b{}'$, unde f este funcția dată inițial.

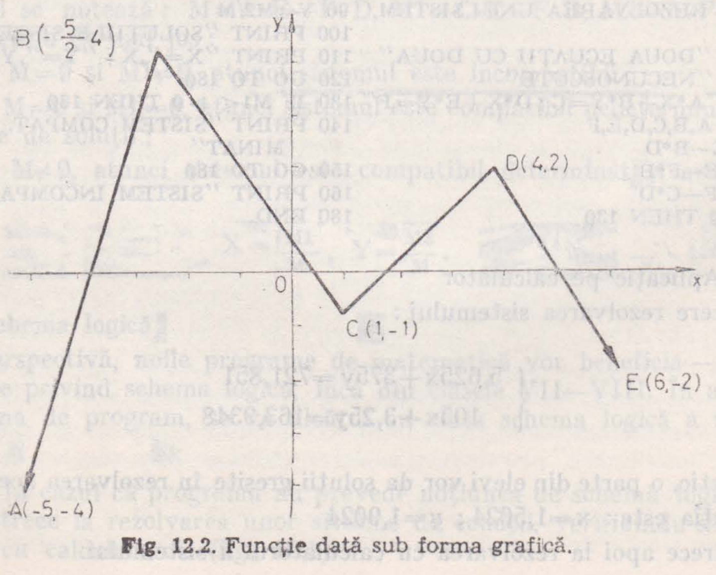


Fig. 12.2. Funcție dată sub forma grafică.

Observaţie. Pentru fiecare din întrebările puse (exceptînd 9°) sînt ataşate cîte două răspunsuri din care unul este corect şi nu prea greu de sesizat. Pentru (9°), explicaţiile suplimentare conţin trei sisteme de axe de coordonate : primul reprezentînd y=f (x) şi y= | f (x) |, al doilea pentru y=f (x) şi y=—| f (x) |, iar al treilea conţine toate cele trei grafice în ordine (linie plină, punctat şi liniuţe). Elevul îl subliniază cu o linie pe cel corect şi predă lucrarea profesorului. În tema de acasă, elevul trebuie să răspundă la întrebările de mai sus, motivînd care sînt elementele de care n-a ţinut seama în cazul răspunsurilor incorecte. Acest mod de lucru îl va stimula să înţeleagă funcţia de „help" a calculatorului, construind el însuşi „comentariul şi trimiterea la material auxiliar" ....

Tema. Funcţii putere de forma

$$f : \mathbb{R} \to \mathbb{R}, (fx) = x^n, n \in N$$

şi funcţii putere de forma :

$$f : \mathbb{R} \setminus \{\emptyset\} \to \mathbb{R}, f(x) = x^n, n \in Z.$$

(Reluăm o secvenţă din prima parte a unei lecţii de sinteză de la finele clasei a IX-a, sau de la începutul clasei a X-a. Calculatorul este folosit aici de către profesor pentru antrenarea gîndirii „productive" a întregii clase).

1°. Prin intermediul consolei vor fi proiectate pe ecran cîteva grafice corespunzătoare lui $n \in \{0, 1, 2, 3, 4, 5\}$, în două etape :

1°. Cazurile particulare (pentru $n \neq \emptyset$ şi pentru $n = 1$) :

$(n = \emptyset \Rightarrow f(x) = 1)$— dreapta de ecuaţie $y = 1$, paralelă cu (Ox), figura 12.3 ;

$(n = 1 \Rightarrow f(x) = x)$ — dreapta de ecuaţie $y = x$, bisectoarea I, figura 12.4 ;

Se reţine de aici ideea că graficele trec prin punctul P (1, 1).

Fig. 12.3

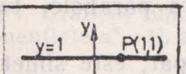
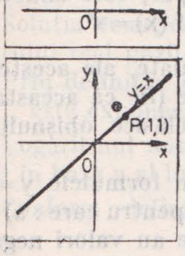


Fig. 12.4



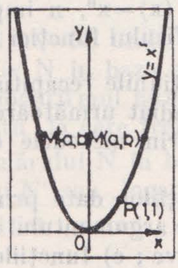


Fig. 12.5

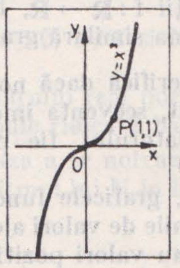


Fig. 12.6

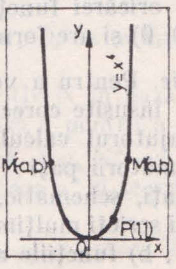


Fig. 12.7

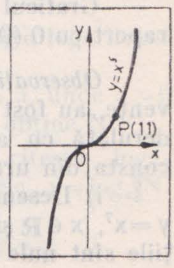


Fig. 12.8

Fig. 12.3. Funcţia $y=1$.
Fig. 12.4. Funcţia $y=x$.
Fig. 12.5. Funcţia $y=x^2$.
Fig. 12.6. Funcţia $y=x^3$.
Fig. 12.7. Funcţia $y=x^4$.
Fig. 12.8. Funcţia $y=x^5$.

2°. În cea de-a doua etapă se proiectează graficele următoarelor funcţii :

$(n = 2 \Rightarrow f(x) = x^2)$ — parabole de ecuaţie $y = x^2$, figura 12.5

$(n = 3) \Rightarrow f(x) = x^3)$ — parabola cubică de ecuaţie $y = x^3$, figura 12.6

$(n = 4) \Rightarrow f(x) = x^4)$ — grafic de tipul parabolei, avînd ecuaţia $y = x^4$, figura 12.7

$(n = 5 \Rightarrow f(x) = x^5)$ — grafic de tipul parabolei cubice, avînd ecuaţia $y = x^5$, figura 12.8

....

$((n = 2p,\ p \in N,\ p \geqslant 1) \Rightarrow (f(x) = x^{2p}))$ — grafic de tip parabolă, de ecuaţie $y = x^{2p}$;

$((n = 2p+1,\ p \in N,\ p \geqslant 1) \Rightarrow (f(x) = x^{2p+1}))$ — grafic de tip parabolă cubică, de ecuaţie $y = x^{2p+1}$.

2. Comentarii ce se pot desprinde din analiza graficelor de funcţii $f(x)=x^n$, $x \in R$, $n \in N$.

1°. Pentru orice n, n-natural, graficul trece prin punctul P (1, 1), şi pentru orice n, $n \in N^*$, graficul trece prin originea axelor, deoarece propoziţiile „$\emptyset=\emptyset^n$" şi „$1=1^n$" sînt adevărate.

2°. Dacă n este număr natural par ($n=2p$, $p>\emptyset$), graficul este simetric în raport cu axa (Oy), deoarece : $f(-x)=(-x)^{2p}=x^{2p}=f(x)$, funcţia f fiind pară. Dacă M (a, b) aparţine graficului funcţiei $f(x)=x^{2p}$, $x \in R$, atunci este adevărată egalitatea $b=a^{2p}$. De asemenea, este adevărată şi egalitatea $b=(-a)^{2p}$; adică, M' (—a, b) aparţine graficului funcţiei date. Fiecărui punct de pe grafic M' (a, b) îi corespunde pe acelaşi grafic simetricul M' (—a, b) în raport cu axa (Oy).

3°. Dacă n este impar, $n=2p+1$, atunci graficul este simetric în raport cu originea axelor de coordonate 0 (∅, ∅). Avem :

$f(-x)=(-x)^{2p+1}=-x^{2p+1}=-f(x)$, funcţia fiind impară.

Dacă M (a, b) aparţine graficului, atunci este adevărată egalitatea $b=a^{2p+1}$ şi, de asemenea, egalitatea $-b=(-a)^{2p+1}$; adică, M' (—a, —b) aparţine graficului. Fiecărui punct de pe grafic M (a, b) îi corespunde pe grafic simetricul M' (—a, —b) în raport cu 0 (∅, ∅).

4°. Graficul oricărei funcţii $f : \mathbb{R} \to \mathbb{R}$, $f(x)=x^n$, n par, este simetric în raport cu axa (Oy), iar pentru $n>\emptyset$ are forma similară parabolei funcţiei $f(x)=x^4$, $x \in R$ (spre exemplu).

Graficul oricărei funcţii $f : \mathbb{R} \to \mathbb{R}$, $f(x)=x^n$, n impar, este simetric în raport cu 0 (∅, ∅) şi are forma similară graficului funcţiei $f(x)=x^5$.

Observaţie. Pentru a verifica dacă noţiunile recapitulate, ale acestei secvenţe, au fost însuşite corect, secvenţa imediat următoare (fie că aceasta este derulată cu ajutorul calculatorului, fie prin metodele clasice obişnuite) ar consta din următorii paşi :

i) Desenaţi, schematic, graficele funcţiilor date prin formulele $y=x^6$ şi $y=x^7$, $x \in \mathbb{R}$ şi scrieţi mulţimile de valori ale argumentului pentru care : a) funcţiile sînt nule ; b) funcţiile au valori pozitive ; c) funcţiile au valori negative.

ii) Pentru funcţia dată prin formula $f(x)=x^{16}$, $x \in \mathbb{R}$, comparaţi f (2) şi f (—2) ; găsiţi f (—a) ştiind că f (a) =32.

iii) Pentru funcţia dată prin formula $f(x)=x^{19}$, $x \in \mathbb{R}$, comparaţi f (3) şi f (—3) şi găsiţi f (—a) ştiind că f (a) =52.

IV) Fie punctul arbitrar M (a, — b) aparţinînd graficului unei funcţii f. Dacă funcţia este dată printr-una din formulele $y=x^5$ sau $y=x^6$ să se precizeze dacă punctele N (—a, b) şi P (—a, —b) aparţin graficului lui f. Evident, pentru acest ultim punct este necesar un mic comentariu.

*Tema. Logaritmi (definiţie şi proprietăţi bazate pe **definiţie**) — lecţie de comunicare, realizată pentru autoinstruirea automatizată.*

1. Noţiunea de logaritm. Să considerăm ecuaţia exponenţială

$$a^x=N,\ N>\emptyset,\ a>\emptyset,\ a\neq 1. \qquad (1)$$

Se observă că ecuaţia (1) nu are soluţie pentru $N<0$, deoarece

$$f : R \rightarrow (0, \infty),\ f(x)=a^x$$

are valori pozitive.

Pentru $N>0$, ecuaţia (1) are soluţie unic determinată, aceasta decurgînd din proprietatea de bijectivitate a funcţiei exponenţiale. În figura 12.9 respectiv 12.10 este prezentată rezolvarea grafică a ecuaţiei $a^x=N$, $a>1$ (respectiv $0<a<1$).

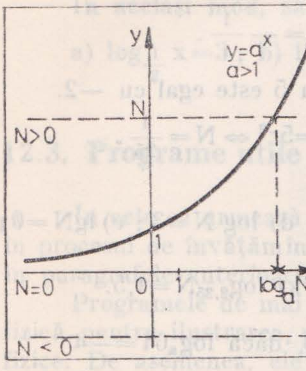


Fig. 12.9. Graficul funcţiei exponenţiale $a^x=N$, $a>1$.

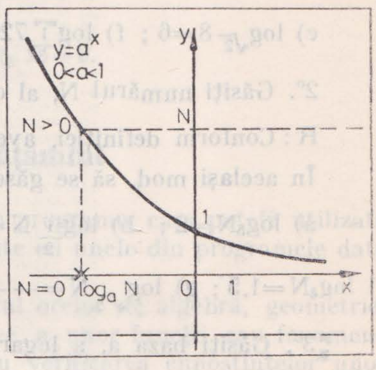


Fig. 12.10. Graficul funcţiei exponenţiale $a^x=N$, $0<a<1$.

Soluţia ecuaţiei $a^x=N$, unde $a>0$, $a\neq 1$ şi $N>0$, se numeşte logaritmul numărului real pozitiv N în baza a.

Prin definiţie, logaritmul numărului real pozitiv N în baza a, unde $a>0$ şi $a\neq 1$, este exponentul la care trebuie ridicată baza a pentru a obţine N.

Logaritmul numărului N în baza a se notează „$\log_a N$", se citeşte „logaritmul în baza a al lui N" sau „logaritmul lui N în baza a" şi se scrie „$x=\log_a N$".

Conform definiţiei avem :

$$\log_2 16=\log_2 2^4=4\ ;\ \log_3 \frac{1}{243}=\log_3\left(\frac{1}{3}\right)^5=\log_3 3^{-5}=-5.$$

Avem deci relaţia : $\log_a a^n=n$.

Din (1) şi (2) rezultă egalitatea : $a^{\log_a N}=N$, $N>0$.

Dacă în (1) punem $x=1$, atunci $a^1=a$ şi conform cu (2) avem relaţia : $\log_a a=1$.

Dacă baza a este 10, atunci în loc de „$\log_{10} N$" se scrie „lg N" şi se citeşte „logaritmul lui N în baza 10" sau „logaritmul zecimal al lui N"; iar dacă baza a este numărul iraţional $e=2{,}71828\ldots$, atunci în loc de $\log_e N$ se scrie „lnN" şi se citeşte „logaritmul natural al numărului N".

2. Aplicaţii privind folosirea definiţiei şi a proprietăţilor bazate pe ea. (Pentru a nu mai prezenta răspunsurile şi comentariile corespunzătoare lor, fiecare grupă de exerciţii începe cu un exerciţiu model, rezolvarea celorlalte fiind analoagă modelului).

1°. Dovediţi că $\log_5 \frac{1}{125} = -3$.

R : Conform definiţiei, avem : $5^{-3} = \frac{1}{125}$ (egalitate adevărată)

În acelaşi mod, să se arate că egalităţile următoare sînt adevărate :

a) $\log_3 81 = 4$; b) $\log_4 2 = \frac{1}{2}$; c) $\lg \frac{1}{10} = -1$; d) $\log_2 \frac{1}{64} = -1$;

e) $\log_{\sqrt{2}} 8 = 6$; f) $\log_{\frac{1}{3}} 729 = -6$; g) $\log_{81} \frac{1}{3} = -\frac{1}{4}$.

2°. Găsiţi numărul N, al cărui logaritm în baza 5 este egal cu -2.

R : Conform definiţiei, avem $\log_5 N = -2 \Leftrightarrow N = 5^{-2} \Leftrightarrow N = \frac{1}{25}$.

În acelaşi mod, să se găsească N ştiind că :

a) $\log_5 N = 2$; b) $\log_{\frac{1}{2}} N = -1$; c) $\lg N = -2$; d) $\log N = 3$; e) $\lg N = 0$;

f) $\log_4 N = 1{,}5$; g) $\log_{\sqrt{2}} N = -\frac{2}{3}$; h) $\log_{64} N = \frac{5}{6}$; k) $\log_{2{,}25} N = 1{,}5$.

3°. Găsiţi baza a, a logaritmului numărului 64, dacă $\log_a 64 = -4$

R : $\log_a 64 = -4 \Leftrightarrow a^{-4} = 64 \Leftrightarrow a = (64)^{-\frac{1}{4}} = (2^6)^{-\frac{1}{4}} = 2^{-\frac{3}{2}} = \frac{1}{\sqrt{2^3}} = \frac{1}{2\sqrt{2}}$.

În acelaşi mod, să se găsească baza a ştiind că :

a) $\log_a 27 = 3$; b) $\log_a 1000 = 1{,}5$; c) $\log_a 27 = -6$; d) $\log_a 64 = \frac{6}{7}$;

e) $\log_a 4\sqrt{2} = -2{,}5$; f) $\log_a (27\sqrt[3]{3}) = \frac{5}{6}$; g) $\log_a \frac{1}{9} = -1$;

h) $\log_a \sqrt[3]{3} = \frac{1}{6}$.

4°. Găsiţi logaritmul numărului 8 în baza $\sqrt{2}$.

R : $\log_{\sqrt{2}} 8 = x \Leftrightarrow (\sqrt{2})^x = 8 \Leftrightarrow 2^{\frac{x}{2}} = 2^3 \Leftrightarrow \frac{x}{2} = 3 \Leftrightarrow x = 6$. Deci $\log_{\sqrt{2}} 8 = 6$.

În acelaşi mod, să se determine numerele :

a) $\log_3 \frac{1}{9}$; b) $\lg 0{,}01$; c) $\log_{\frac{1}{5}} 25$; d) $\log_{\frac{1}{\sqrt{3}}} 27$; e) $\log_{0.2} 25$;

f) $\log_4 (8\sqrt{2})$; g) $\log_{27} \frac{\sqrt{3}}{9}$.

5°. Găsiţi valoarea expresiei $10^{\lg 12}$.

R : $10^{\lg 12} = 10^{\log_{10} 12} = 12$.

În acelaşi mod, să se găsească valorile expresiilor :

a) $10^{-\lg 5}$; b) $10^{\lg 100}$; c) $10^{\lg 0{,}1}$; d) $100^{\lg 7}$.

6°. Care din expresii are sens?

a) $\log_4 \emptyset$; b) $\lg \emptyset{,}\emptyset1$; c) $\sqrt{\lg \emptyset{,}9}$; d) $\lg(\lg 9{,}6)$; e) $\log_3(-7)^2$;

f) $\log_3(-3)^3$; g) $\log_4(-4)^{-4}$; h) $\sqrt{\lg 2{,}5}$.

7. Rezolvaţi ecuaţia $\log_3 x = -1$

R : $\log_3 x = -1 \Leftrightarrow x = 3^{-1} \Leftrightarrow x = \frac{1}{3}$.

În acelaşi mod, să se rezolve ecuaţiile :

a) $\log_{\frac{1}{2}} x = 3$; b) $\log_{0,3} x = 2$; c) $\log_3(\log_5 x) = \emptyset$.

12.3. Programe utile în procesul de învăţămînt

În cele ce urmează se prezintă o serie de programe, care pot fi utilizate în procesul de învăţămînt. Ele pot fi completate cu unele din programele date în paragrafele anterioare ale acestui capitol.

Programele de mai jos sînt utile în cadrul orelor de algebră, geometrie, fizică, pentru ilustrarea, prin reprezentări grafice, a unor funcţii sau fenomene fizice. De asemenea, ele pot fi folosite pentru verificarea cunoştinţelor unor grupuri de elevi, în privinţa despărţirii cuvintelor în silabe, a cunoaşterii reşedinţelor de judeţe sau pe alte teme. La sfîrşitul examinării candidaţii sînt ordonaţi automat după mediile obţinute.

12.4. Program pentru trasarea cercului trigonometric.

Programul trasează cercul trigonometric şi calculează funcţiile sin, cos tg, pentru un unghi dat. Unghiul se dă sub forma unui multiplu k, de 5 grade — (PI/36).

```
5 INIT F
10 REM"CERC TRIGONOMETRIC
12 REM"K=MULTIPLU DE 5 GRADE
15 INPUT K
20 Q=K*PI/36
25 MOVE 50,50
30 FOR J=0 TO 2*PI STEP PI/36
40 DRAW 25*COS(J)+50,25*SIN(J)+50
50 NEXT J
60 MOVE 50,50
70 RDRAW -25,0
80 RMOVE 25,25
90 RDRAW 0,-50
100 MOVE 50,50
110 RDRAW 25*COS(Q),25*SIN(Q)
120 RDRAW 0,-25*SIN(Q)
125 P=Q*180/PI
130 PRINT AT(17,20)"A(1,0)"
140 PRINT AT(8,9)"B(0,1)"
150 PRINT AT(17,1)"C(-1,0)"
160 PRINT AT(35,9)"D(0,-1)"
165 PRINT AT(25,1)"U=";U
170 PRINT AT(27,1)"Q=";Q
180 PRINT AT(28,1)"SIN(Q)=";SIN(Q)
190 PRINT AT(29,1)"COS(Q)=";COS(Q)
200 PRINT AT(30,1)"TAN(Q)=";TAN(Q)
400 END
```

12.5. Program pentru vizualizarea pozițiilor unor drepte care trec prin originea axelor de coordonate.

Programul are un caracter conversațional, solicitînd inițial numărul k de drepte, care urmează a fi reprezentate, coeficienții unghiulari, M (k), ai acestor drepte și limitele pe orizontală ale ferestrei de afișare (X1, X2).

```
5 INIT P
10 REM"VIZUALIZAREA POZITIEI
20 REM"UNEI DREPTE CE TRECE
30 REM"PRIN O(0,0), IN FUNCTIE
40 REM"DE COEFICIENTUL UNGHIU-
50 REM"LAR M=TAN(U),UNDE
60 REM"Y=M*X
70 REM"VARIABILE UTILIZATE
80 REM"K = NUMAR DE DREPTE
90 REM"M(K) = VECTORUL ALE
100 REM"CARUI COMPONENTE SINT
110 REM"COEFICIENTII UNGHIULARI
120 REM"LIMITELE FERESTREI
130 REM"AFISATE : X1=LIMITA
140 REM"STINGA, X2=LIMITA DREAP
150 REM"TA, Y1 = LIMITA JOS,Y2=
160 REM"LIMITA SUS
200 PRINT "INTRODUCETI K":
210 INPUT K
220 PRINT "INTRODUCETI COEFICIENTII"
225 DIM M(K)
230 FOR I=1 TO K
240 PRINT " M(";I;")=";
250 INPUT M(I)
260 NRXT I
270 PRINT " INTRODUCETI LIMITELE X1,X2"
280 INPUT X1,X2
290 Y1=X1
300 Y2=X2
310 REM"AFISAREA GRAFICULUI
315 INIT P
320 VIEWPORT 40,120,20,100
330 WINDOW X1,X2,Y1,Y2
340 REM"TRASARE AXE SI CONTUR
350 MOVE X!,Y1
360 DRAW X2,Y1
370 DRAW X2,Y2
380 DRAW X1,Y2
390 DRAW X1,Y1
400 MOVE X1,0
410 DRAW X2,0
420 MOVE 0,Y1
430 DRAW 0,Y2
440 PRINT AT (1,20)"Y"
450 PRINT AT(15,30)"X"
500 R=(X2-X1)/20
510 FOR I=1 TO K
515 PRINT AT(2*I,1);"M":I;"=";M(I
520 R1=I*R
530 MOVE X1,M(I)*X1
540 DRAW X2,M(I)*X2
550 MOVE R1,0
555 U=ATN(M(I))
556 IF U=> THEN 600
557 IF U>0 THEN 560

558 U=PI+U
560 FOR J=0 TO U STEP U/10
570 DRAW R1*COS(J),R1*SIN(J)
580 NEXT J
600 NEXT I
700 END
```

12.6. Graficul funcției de gradul doi.

Programul afișează într-o fereastră dată (X1, X2) o familie de parabole de gradul doi (Y=A*X ↑ 2) cu diferiți coeficienți. Limitele verticale ale ferestrei se calculează automat.

```
5 INIT P
10 REM"GRAFICUL FUNCTIEI
20 REM"DE GRADUL DOI
30 REM"Y=A*X^2
40 REM"VARIABILE UTILIZATE
50 REM"K=NUMAR DE PARABOLE
60 REM"A(K)=VECTORUL ALE
70 REM"CARUI ELEMENTE SINT
80 REM"COEFICIENTII LUI X^2
90 REM"LIMITELE FERESTEREI
100 REM"AFISATE : X1=LIMITA
110 REM"STINGA,X2=LIMITA
120 REM"DREAPTA,Y1=LIMITA JOS
130 REM"Y2=LIMITA SUS
140 REM"FEREASTRA SE CONSIDERA
150 REM"SIMETRICA, DECI:
110 REM"ABS(X1)=ABS(X2)=X SI
170 REM"ABS(Y1)=ABS(Y2)=X^2
200 PRINT "INTRODUCETI K";
210 INOUT K
220 PRINT "INTRODUCETI COEFICIENTII";
230 DIM A(K)
240 FOR I=1 TO K
250 PRINT "A(";I;")=";
260 INPUT A(I)
270 NEXT I
280 PRINT "INTRODUCETI X";
290 INPUT X
300 X1=-X
310 X2=X
320 Y1=-X^2
320 Y2=X^2
340 REM"AFISARE GRAFICA
350 INIT P
360 VIEWPORT 40,120,20,100
370 WINDOW X1,X2,Y1,Y2
380 REM"TRASARE AXE SI CONTUR
400 MOVE X1,Y1
410 DRAW X2,Y1
420 DRAW X2,Y2
430 DRAW X1,Y2
440 DRAW X1,Y1
450 MOVE X1,0
460 DRAW X2,0
470 MOVE 0,Y1
480 DRAW 0,Y2
540 PRINT AT(1,20)"Y"
550 PRINT AT(15,30)"X"
560 PRINT AT(1,1);"Y=A*X^2"
600 P=X/20
610 FOR I=1 TO K
615 PRINT AT(2*I,1)"A";I;"=";A(I)
620 MOVE X1,A(I)*X1^2
630 FOR J=X1 TO X2 STEP P
640 DRAW J,A(I)*J^2
650 NEXT J
660 NEXT I
700 END
```

12.7. Graficul funcţiei de gradul N

Programul reprezintă graficele unei familii de parabole (Y=A (k)* X ^ N), unde A (k) este coeficientul parabolei cu numărul k, iar N>=0. Programul solicită numărul k de parabole, coeficienţii A (k) şi limitele orizontale X1, X2 ale ferestrei de afişare.

```
5 INIT P
10 REM"GRAFICUL FUNCTIEI
20 REM"DE GRADUL N
30 REM"Y=A*X^N
40 REM"VARIABILE UTILIZATE:
50 REM"K=NUMAR DE PARABOLE
60 REM"A(K)=VECTORUL ALE
70 REM"CARUI ELEMENTE SINT
80 REM"COEFICIENTII LUI X^N
90 REM"LIMITELE FERESTREI
100 REM"AFISATE:X1=LIMITA
110 REM"STINGA,X2=LIMITA
120 REM"DREAPTA,Y1=LIMITA JOS
130 REM"Y2=LIMITA SUS
140 REM"FEREASTRA SE CONSIDERA
150 REM"SIMETRICA, DECI
160 REM"ABS(X1)=ABS(X2)=X SI
170 REM"ABS(Y1)=ABS(Y2)=X^2
200 PRINT "INTRODUCETI K";
210 INPUT K
220 PRINT "INTRODUCETI COEFICIENTII"
230 DIM A(K)
235 DOM N(K)
240 FOR I= 1 TO K
250 PRINT "A(";I;")=";
260 INPUT A(I)
265 PRINT "N(";I;")=";
266 INPUT N(I)
270 NEXT I
280 PRINT "INTRODUCETI X";
290 INPUT X
300 X1=-X
310 X2=X
320 Y1=-X^2
330 Y2=X^2
340 REM"AFISARE GRAFICA
350 INIT P
360 VIEWPORT 40,120,20,100
370 WINDOW X1,X2,Y1,Y2
380 REM"AXE SI CONTUR
400 MOVE X1,Y1
410 DRAW X2,Y1
420 DRAW X2,Y2
430 DRAW X1,Y2
440 DRAW X1,Y1
450 MOVE X1,0
460 DRAW X2,0
470 MOVE 0,41
480 DRAW 0.42
540 PRINT AT(1,20)"Y"
550 PRINT AT(15,30)"X"
560 PRINT AT(1,1);"Y=A*X^N"
600 P=X/20
610 FOR I= 1 TO K
615 PRINT AT(2*I,1)"A";I;"=";A(I)

616 PRINT AT(2*I+1,1)"N";I;"=";N(I.
625 MOVE X1,A(I)*X1^N(I)
630 FOR J=X1 to X2 STEP P
640 DRAW J,A(I)*J^N(I)
650 NEXT J
660 NEXT I
670 PRINT "DORITI ALT X?, DA=1"
675 INPUT D
680 IF D=1 THEN 280
700 END
```

12.8. Graficul funcției logaritmice.

Programul afișează graficul funcției logaritmice în bazele 2, 3, 5 și 10. Inițial se solicită limitele orizontale de afișare.

```
5 INIT F
10 REM "GRAFICUL FUNCTIEI
20 REM "LOGARITMICE IN
30 REM "BAZELE : 2,3,5,10
35 REM "LIMITELE INTERVALULUI
40 REM "DE AFISARE
45 REM "X1 = LIMITA STINGA
50 REM "X2 = LIMITA DREAPTA
55 REM "Y1 = LIM.JOS. Y2 = LIM. SUS
60 REM "LIMITA X1 VA FI CALCULATA
75 REM "PENTRU Y = LOG(X)/LOG(10)
80 REM "DACA X >= 1 LIMITA Y2 SE
VA CALCULA
85 REM "PENTRU Y - LOG(X)/LOG(2)
90 PRINT "INTRODUCETI LIMITELE"
95 PRINT "X1 SI X2"
100 PRINT "X1 = ",
110 INPUT X1
120 PRINT "X2 = ",
130 INPUT X2
140 IF X1 <= 0 THEN 500
150 Y1 = LOG(X1)/LOG(2
160 IF X2< 1 THEN 190
170 Y2 = LOG(X2)/LOG(2)
180 GOTO 200
190 Y2 = LOG(X2)/LOG(10
200 REM "AFISARE GRAFICA
210 INIT P
220 VIEWPORT 40,120,0.70
230 WINDOW X1,X2,Y1,Y2
240 REM "TRASARE AXE SI CONTUR
250 MOVE X1,Y1
260 DRAW X2,Y1
270 DRAW X2,Y2
280 DRAW X1,Y2
290 DRAW X1,Y1
300 MOVE X1,0
310 DRAW X2,0
320 MOVE 0,Y1
330 DRAW 0,Y2
340 PRINT AT(10,10)"Y"
350 PRINT AT(32,30)"X"
360 PRINT AT(1,1);"Y=LOG(X)/LOG(2)
370 PRINT AT(2,1);"Y=LOG(X)/LOG(3)
380 PRINT AT(3,1);"Y=LOG(X)/LOG(5)"
390 PRINT AT(4,1);"Y=LOG(X)/LOG(10)
400 PRINT AT(5,1);"X1=";X1
402 PRINT AT(6,1);"X2=";X2
404 PRINT AT(7,1);"Y1=";Y1
406 PRINT AT(8,1);"Y2=";Y2
410 K = 2
415 GOSUB 600
420 K=3
425 GOSUB 600
430 K=5
435 GOSUB 600
440 K = 10
450 GOSUB 600
470 PRINT"DORITI ALTE LIMITE ? DA=1"
480 INPUT D
490 IF D=1 THEN 90
495 GOTO 700
500 PRINT "LIMITA X1 INCORECTA"
510 STOP
600 P=(X2-X1)/100
610 FOR J = X1 TO X2 STEP P
620 DRAW J.LOG(J)/LOG(K)
630 NEXT J
640 MOVE X1,Y1
650 RETURN
700 END
```

12.9. Program pentru studiul aruncării corpurilor sub un unghi dat

Programul solicită unghiul sub care are loc aruncarea, viteza iniţială şi viteza vîntului.

Pentru încadrarea graficului în limitele ecranului, programul execută o scalare pe verticală, calculînd iniţial înălţimea maximă la care ajunge corpul. În acest scop solicită numărul de paşi pentru a calcula valorile parabolei în punctele respective, reţinînd valoarea maximă.

Se afişează înălţimea maximă şi bătaia.

```
1 INIT P
2 REM"PROGRAM PENTRU STUDIUL ARUN-
CARII CORPURILOR
3 PRINT "INTRODUCETI UNGHIUL , U,V.INIT:V,V.VINT:
W"
4 INPUT U,V,W
5 PRINT "INTRODUCETI NUMARUL DE PASI N"
7 INPUT N
30 INIT P
31 L=PI*U/180
32 K=V*COS(L)-W
33 B=(V*SIN(L))/K
34 A=-4.905/K^2
35 X2=K*V*SIN(L)*2/9.81
36 R2=(1/19.62)*(V*SIN(L))^2
37 PRINT AT(28,1)"H.MAX = ";R2
39 PRINT AT(29,1)"X.MAX =";X2
45 GOSUB 1000
50 STOP
60 END
1000 IF X1< X2 THEN 1030
1010 PRINT "LIMITE GRESITE"
1020 STOP
1030 H=(X2-X1)/N
1040 VIEWPORT 40,100,20,100
1050 X1=0
1070 R1=0
1100 IF X2>=R2 THEN 1140
1110 X2=R2
1120 GOTO 1190
1140 R2=X2
1190 WINDOW X1,X2,R1,R2
1200 MOVE X1,0
1201 DRAW X2,0
1202 MOVE 0,R1
1203 DRAW 0,R2
1210 X=X1
1220 GOSUB 1500
1230 MOVE X,F
1240 FOR I=1 TO N
1250 X=X+H
1260 GOSUB 1500
1270 DRAW X,F
1280 NEXT I
1290 RETURN
1500 P=A*X^2+B*X
1510 RETURN
```

12.10. Calculul punctului de intersecţie a două drepte

O dreaptă poate fi definită prin specificarea coordonatelor unui punct M (x, y) şi a unghiului pe care îl face cu direcţia pozitivă a axei Ox. Fie două drepte D1 şi D2 de ecuaţii parametrice:

$$D1: \begin{cases} X = X_1 + L_1 \cos U_1 \\ Y = Y_1 + L_1 \sin U_1 \end{cases}$$

$$D2: \begin{cases} X = X_2 + L_2 \cos U_2 \\ Y = Y_2 + L_2 \sin U_2 \end{cases}$$

Fie ($X_\emptyset$, $Y_\emptyset$) coordonatele punctului de intersecţie. Înlocuind în ecuaţiile parametrice, se obţine sistemul:

$$X_\emptyset = X_1 + L_1 \cos U_1$$

$$Y_\emptyset = Y_1 + L_1 \sin U_1$$

$$X\emptyset = X_2 + L_2 \cos U_2$$

$$Y\emptyset = Y_2 + L_2 \sin U_2$$

De aici rezultă :

$$L_1 \cos U_1 - L_2 \cos U_2 = X_2 - X_1$$

$$L_1 \sin U_1 - L_2 \sin U_2 = Y_2 - Y_1$$

De unde : $L_1 = ((X_1 - X_2) \sin U_2 - (Y_1 - Y_2) \cos U_2)/\sin (U_1 - U_2)$.

Avînd determinat L_1, coordonatele punctului de intersecţie vor fi :

$$X\emptyset = X_1 + L_1 \cos U_1$$

$$Y\emptyset = Y_1 + L_1 \sin U_1$$

```
1 REM "******************************"
2 REM "* CALCULUL PUNCTULUI DE      *"
3 REM "* INTERSECTIE A DOUA DREPTE *"
4 REM "******************************"
5 DIM D$(1)
6 REM "CITIREA DATELOR"
7 PRINT "INTRODUCETI PARAMETRII DREPTELOR"
10 PRINT "X1,Y1,U1(GRADE)"
15 INPUT X1,Y1,U1
20 PRINT "X2,Y2,U2(GRADE)"
25 INPUT X2,Y2,U2
27 REM "TESTARE PARALELISM DREPTE"
30 N=ABS((U1-U2)/180)
35 IF INT(N)<>N THEN 50
40 PRINT "DREPTELE SINT PARALELE"
45 STOP
47 REM "CALCULUL COORDONATELOR"
48 REM "PUNCTULUI DE INTERSECTIE"
50 U1=U1*PI/180
55 U2=U2*PI/180
60 D1=(X1-X2)*SIN(U2)-(Y1-Y2)*COS(U2)
65 D=SIN(U1-U2)
70 L=D1/D
75 XO=X1+L*COS(U1)
80 YO=Y1+L*SIN(U1)
85 PRINT "COORDONATELE PUNCTULUI"
86 PRINT "DE INTERSECTIE SINT:"
90 PRINT "XO=";XO
95 PRINT "YO=";YO
100 PRINT "DORITI AFISARE GRAFICA? (DA,NU)
105 INPUT D$(1)
110 IF D$(1)='D' THEN 120
115 STOP
118 REM "DEFINIREA SPATIULUI DE AFISAT"
120 T1=ABS(XO)
125 T2=ABS(YO)
130 IF T1>T2 THEN140
135 T1=T2
140 T1=2*T1
145 IF T1<>O THEN 149
146 T1=20
149 INIT
150 VIEWPORT 10,90,10,90
155 WINDOW -T1,T1,-T1,T1
157 REM "TRASAREA AXELOR"
160 MOVE -T1,O
165 DRAW T1,O
170 MOVE O,-T1
175 DRAW O,T1
178 REM "TRASAREA DREPTELOR"
180 MOVE X1-2*T1*COS(U1),Y1-2*T1*SIN(U1)
185 DRAW X1+2*T1*COS(U1),Y1+2*T1*SIN(U1)
190 MOVE X2-2*T2*COS(U2),Y2-2*T2*SIN(U2)
195 DRAW X2+2*T2*COS(U2),Y2+2*T2*SIN(U2)
200 MOVE XO,YO
205 DRAW XO,O
210 MOVE XO,YO
215 DRAW O,YO
220 END
```

Dacă dreptele sînt paralele numitorul expresiei pentru calculul lui L_1 este nul. Deci se va testa condiția de neparalelism U1—U2≠Kπ.

Programul funcționează astfel :

1. Citește parametrii celor două drepte ; X1, Y1, U1 și X2, Y2, U2.
2. Testează dacă cele două drepte sînt paralele. Dacă dreptele sînt paralele se oprește execuția programului.
3. Calculează coordonatele punctului de intersecție și apoi le afișează.
4. Afișează grafic (eventual) rezultatul.

La definirea spațiului de afișat în linia 155, s-a avut în vedere ca originea și axele sistemului de coordonate să apară pe ecran.

12.11. Calculul punctelor de intersecție a două cercuri

Se consideră cercurile definite prin centru și rază : C1 (X1, Y1, R1) și C2 (X2, Y2, R2). Distanța dintre centrele celor două cercuri este :

$$D=\sqrt{(X2-X1)^2+(Y2-Y1)^2}$$

Se va nota G=X2—X1 și H=Y2—Y1.

Pentru a se intersecta, distanța dintre cercuri trebuie să fie mai mică decît suma razelor și mai mare decît diferența lor : $|R2-R1| \leqslant D \leqslant R1+R2$.

În cazul egalității, cercurile vor fi tangente. Se notează cu U1 unhiul format de dreapta ce unește centrele cercurilor cu orizontala și U2 unghiul format de această dreaptă cu raza primului cerc dusă în punctul de intersecție dorit. Coordonatele punctului de intersecție vor fi :

$$X=X1+R1*\cos(U1+F*U2)$$

$$Y=Y1+R1*\sin(U1+F*U2)$$

Unde : F=1 pentru punctul din stînga dreptei ce unește centrele (privit din C1) și —1 pentru cel din dreapta.

Pentru calculul unghiurilor U1 și U2 folosim relațiile :

$$U1=ATN(G/H), \text{ sau } \pi/2 \text{ dacă } H=0$$

și

$$U2=\arccos(R1^2+D^2-R2^2)/(2*R1*D)$$

însă arccos va fi calculat prin funcția ATN.

Programul funcționează în felul următor :

1. Citește coordonatele centrului și raza X1, Y1, R1 pentru primul cerc.
2. Citește coordonatele centrului și raza X2, Y2, R2 pentru al doilea cerc.

3. Citeşte valoarea lui F (1 =punctul din stînga, —1 =cel din dreapta)
4. Calculează valorile G, H şi D. Dacă nu e verificată condiţia : $|R2-R1| \leqslant D \leqslant R1+R2$ execuţia se opreşte.
5. Calculează unghiurile U1 şi U2.
6. Calculează coordonatele X, Y ale punctului de intersecţie dorit.
7. Afişare grafică (eventuală) a rezultatului.
8. Stop.

```
002 REM "**************************************************"
005 REM "* CALCULUL PUNCTELOR DE INTERSECTIE A DOUA CERCURI *"
010 REM "**************************************************"
012
015 DIM R$(1)
017
020 PRINT "INTRODUCETI COORDONATELE CENTRULUI SI RAZA"
025 PRINT "PENTRU PRIMUL CERC(X1,Y1,R1)";
030
035 INPUT X1,Y1,R1
040 IF R1<=0 THEN 070
045
050 PRINT "PENTRU AL DOILEA CERC(X2,Y2,R2)";
055
060 INPUT X2,Y2,R2
065 IF R2>0 THEN 095
070 PRINT "RAZA NEGATIVA SAU NULA!"
075 GO TO 590
080
085 REM "CALCULUL MARIMILOR G,H,D"
090
095 G=X2-X1
100 H=Y2-Y1
105 D=SQR(G*G+H*H)
110
115 REM "VERIFICREA CONDITIILOR DE INTERSECTIE"
120
125 IF D<>0 THEN 145
130 IF R1<>R2 THEN 155
135 PRINT "CERCURI IDENTICE!"
140 GO TO 320
145 IF D>R1+R2 THEN 155
150 IF D>=ABS(R1-R2) THEN 180
155 PRINT "CERCURILE NU SE INTERSECTEAZA!"
160 GO TO  20
165
170 REM "CALCULUL UNGHIULUI U1"
175
180 U1=PI/2
185 IF G<>0 THEN 205
190 IF H>0 THEN 235
195 U1=-U1
200 GO TO 235
205 U1=ATN(H/G)
210 IF G>0 THEN 235
215 U1=U1+PI
220
225 REM "CALCULUL UNGHIULUI U2"
230
235 L1=(R1*R1+D*D-R2*R2)/(2*R1*D)
240 U2=PI/2
245 IF L1=0 THEN 275
250 L2=SQR(1-L1*L1)/L1
255 U2=ATN(L2)
260 IF L1>0 THEN 275
265 U2=U2+PI
270
275 PRINT "PUNCTELE DE INTERSECTIE AU COORDONATELE:"
280
285 F=1
```

```
290 X=X1+R1*COS(U1+F*U2)
295 Y=Y1+R1*SIN(U1+F*U2)
300 PRINT "X=";X,"Y=";Y
305 F=-F
310 IF F<>1 THEN 290
315
320 PRINT "DORITI AFISARE GRAFICA(D/N)";
325
330 INPUT R$
335 IF R$<>"D" THEN 590
340
345 REM "STABILIREA CONDITIILOR DE AFISARE"
350
355 INIT
360 VIEWPORT 40,140,0,100
365 A=X1
370 B=X2
375 GO SUB 545
380 T1=T
385 A=Y1
390 B=Y2
395 GO SUB 545
400 IF T>=T1 THEN 410
405 T=T1
410 WINDOW -T,T,-T,T
411 MOVE -T,-T
412 DRAW T,-T
413 DRAW T,T
414 DRAW -T,T
415 DRAW -T,-T
416
420 REM "TRASAREA CERCURILOR"
425
430 J=2*PI+PI/10
435 K=PI/10
440 MOVE X1+R1,Y1
445 FOR I=0 TO J STEP K
450     DRAW X1+R1*COS(I),Y1+R1*SIN(I)
455 NEXT I
460 MOVE X2+R2,Y2
465 FOR I=0 TO J STEP K
470     DRAW X2+R2*COS(I),Y2+R2*SIN(I)
475 NEXT I
480
485 REM "TRASAREA AXELOR"
490
495 MOVE -T,0
500 DRAW T,0
505 MOVE 0,-T
510 DRAW 0,T
515 MOVE -T,-T
520 GO TO 590
525
530 REM "SUBRUTINA DE CALCUL A COORDONATEI MAXIME"
535 REM "PE O DIRECTIE DATA"
540
545 T=ABS(A-R1)
550 IF T>=ABS(B-R2) THEN 560
555 T=ABS(B-R2)
560 IF T>=ABS(A+R1) THEN 570
565 T=ABS(A+R1)
570 IF T>=ABS(B+R2) THEN 580
575 T=ABS(B+R2)
580 RETURN
585
590 PRINT "DORITI ALTE CERCURI(D/N)";
595
600 INPUT R$
605 IF R$="D" THEN 020
610 END
```

12.12. Calculul tangentelor dintr-un punct la un cerc

Fie o dreaptă D de ecuaţii parametrice :

$$X = X_1 + L \cos T$$

$$Y = Y_1 + L \sin T$$

unde : X_1, Y_1 sînt coordonatele unui punct precizat al dreptei, T direcția dreptei, iar L este distanța de la punctul (X_1, Y_1) la punctul curent.

Fie un cerc C, de centru (X_0, Y_0) și rază R, de ecuație :

$$(X-X\emptyset)^2+(Y-Y\emptyset)^2=R^2$$

Pentru a se obține tangentele din punctul (X_1, Y_1) la cercul C, se vor înlocui X și Y dați de ecuația dreptei D în ecuația cercului. Se obține :

$$L^2+2L\,(H\cos T+V\sin T)+V^2+H^2-R^2=\emptyset \quad (1)$$

unde s-a notat : $H=X_1-X\emptyset$ și $V=Y_1-Y_0$.

Se presupune că punctul (X_1, Y_1) este exterior cercului C, deci : $\sqrt{H^2+V^2}>R^2$. Condiția ca dreapta D să fie tangentă la cerc este ca ecuația (1) de gradul doi în L, să aibă soluție unică. Pentru aceasta trebuie ca :

$$(H\cos T+V\sin T)^2=V^2+H^2-R^2 \quad (2)$$

În rezolvarea acestei ecuații se disting două cazuri

a) $H\neq 0$. Din (2) se obține :

$$\cos T+\frac{V}{H}\sin T=\pm\frac{\sqrt{V^2+H^2-R^2}}{H}$$

Fie $F=\operatorname{arctg}\left(\frac{V}{H}\right)$. Înlocuind mai sus se obține :

$$\cos(T-F)=\pm\sqrt{\frac{V^2+H^2-R^2}{V^2+H^2}},$$

$$\tan(T-F)=\pm\frac{R}{\sqrt{V^2+H^2-R^2}}$$

și deci $T_{1,2}=\operatorname{arctg}\frac{V}{H}+\operatorname{arctg}\frac{\pm R}{\sqrt{V^2+H^2-R^2}}$.

b) $V\neq 0$. Din ecuația (2), împărțind cu V rezultă :

$$\sin(T+F)=\pm\sqrt{\frac{V^2+H^2-R^2}{H^2+V^2}},$$

unde F este unghiul pentru care $\tan F=\frac{H}{V}$.

Procedînd ca la cazul precedent, se obține :

$$T_{1,2}=-\operatorname{arctg}\frac{H}{V}+\operatorname{arctg}\left(\pm\frac{\sqrt{V^2+H^2-R^2}}{R}\right).$$

Se observă că în ambele cazuri se obțin două unghiuri T, corespunzătoare celor două tangente la cerc. Avînd direcțiile tangentelor determinate printr-una din cele două metode, soluția ecuației (1) corespunzătoare unghiurilor

T_1 şi T_2 va fi : $L_{1,2}=-(H\cos T_{1,2}+V\sin T_{1,2})$. Coordonatele punctelor de tangenţă se vor obţine înlocuind valorile obţinute pentru T şi L în ecuaţia dreptei D. S-au prezentat cele două cazuri de rezolvare a ecuaţiei (2), deoarece este posibil ca punctul (X_1, Y_1) să se afle pe aceeaşi verticală sau orizontală cu centrul cercului.

În aceste condiţii va rezulta $H=\emptyset$ sau $V=\emptyset$. (Evident, H şi V nu pot fi simultan nuli). Programul va testa dacă $H\neq\emptyset$. În acest caz, direcţiile tangentelor vor fi determinate prin prima metodă ; altfel se va alege cealaltă soluţie.

Funcţionarea programului :

1. Citirea datelor

a) centrul şi raza cercului (X∅, Y∅, R)

b) punctul din care se duc tangentele (X1, Y1)

2. Testează dacă punctul dat este în interiorul sau pe conturul cercului. În caz afirmativ se va afişa un mesaj corespunzător şi se va opri execuţia.

3. Se determină direcţiile celor două tangente şi coordonatele punctelor de tangenţă şi se afişează rezultatele.

4. Afişează (după dorinţă) grafic soluţia găsită.

```
1 REM "************************************"
2 REM "*  CALCULUL DREPTELOR DE TANGENTA   *"
3 REM "* DINTR-UN PUNCT DAT LA UN CERC DAT *"
4 REM "************************************"
5 REM "CITIREA DATELOR"
6 DIM D$(1)
7 PRINT "INTRODUCETI COORDONATELE CENTRULUI"
10 PRINT "SI RAZA CERCULUI (X0,Y0,R)"
15 INPUT X0,Y0,R
20 PRINT "INTRODUCETI COORDONATELE PUNCTULUI (X1,Y1)"
25 INPUT X1,Y1
30 LET H=X1-X0
35 LET V=Y1-Y0
40 LET D=SQR(V*V+H*H)
42 REM "SE TESTEAZA DACA PUNCTUL (X1,Y1)"
43 REM "ESTE IN INTERIORUL CERCULUI"
45 IF D>R THEN 75
50 IF D=R THEN 65
55 PRINT "PUNCTUL ESTE INTERIOR CERCULUI"
60 STOP
65 PRINT "PUNCTUL ESTE PE CERC"
70 STOP
72 REM "PUNCT EXTERIOR CERCULUI"
73 REM "CALCULUL PUNCTELOR DE TANGENTA"
75 LET R1=SQR(V*V+H*H-R*R)
80 IF H=0 THEN 100
85 LET T1=ATN(V/H)+ATN(R/R1)
90 LET T2=ATN(V/H)-ATN(R/R1)
95 GOTO 110
100 LET T1=-ATN(H/V)+ATN(R1/R)
105 LET T2=-ATN(H/V)-ATN(R1/R)
110 LET L1=-(H*COS(T1)+V*SIN(T1))
115 LET L2=-(H*COS(T2)+V*SIN(T2))
120 LET Z1=X1+L1*COS(T1)
125 LET W1=Y1+L1*SIN(T1)
130 LET Z2=X1+L2*COS(T2)
135 LET W2=Y1+L2*SIN(T2)
140 PRINT "COORDONATELE PUNCTELOR DE TANGENTA SINT:"
145 PRINT "XA=";Z1,"YA=";W1
150 PRINT "XB=";Z2,"YB=";W2
155 PRINT "DORITI AFISARE GRAFICA ? (DA,NU)"
160 INPUT D$(1)
165 IF D$(1)='D'THEN 290
167 REM "DEFINIREA SPATIULUI UTILIZATOR"
168 REM "SI A PORTIUNII DE ECRAN PENTRU AFISARE"
170 LET L1=ABS(X0)
175 IF L1>ABS(Y0) THEN 185
180 LET L1=ABS(Y0)
185 IF L1>ABS(X1) THEN 195
190 LET L1=ABS(X1)
195 IF L1>ABS(Y1) THEN 205
200 LET L1=ABS(Y1)
205 IF L1<>0 THEN 215
210 L1=20
215 LET L1=2*L1
220 INIT
225 VIEWPORT 10,90,10,90
230 WINDOW -L1,L1,-L1,L1
232 REM "TRASAREA CERCULUI"
235 MOVE X0+R,Y0
240 FOR I=0 TO 2*PI+PI/60 STEP PI/60
245     DRAW X0+R*COS(I),Y0+R*SIN(I)
250 NEXT I
253 REM "TRASAREA TANGENTELOR"
255 MOVE Z1,W1
260 DRAW X1,Y1
265 DRAW Z2,W2
268 REM "TRASAREA AXELOR DE COORDONATE"
270 MOVE -L1,0
275 DRAW L1,0
280 MOVE 0,-L1
285 DRAW 0,L1
290 END
```

12.13. Calcul cu polinoame

Se va studia împărţirea unui polinom : $a_1x^n+a_2x^{n-1}+\ldots+a_{n+1}$ printr-un polinom ireductibil şi unitar peste R, anume cazul împărţirii prin $x+a$ şi x^2+px+q. Coeficienţii polinomului cît : $b_1x^n+b_2x^{n-1}+\ldots+b_{n+1}$, vor fi determinaţi prin identificare în relaţia :

$$(b_1x^n+b_2x^{n-1}+\ldots+b_{n+1})(x+a)+b_{n+2}=a_1x^n+a_2x^{n-1}+\ldots+a_{n+1}$$

unde cu b_{n+2} s-a notat restul. Rezultă :

$$b_1=0 \text{ şi } b_{i+1}=a_i-ab_i, \text{ pentru } i=1, 2, \ldots, n+1.$$

Programul de calcul al coeficienţilor cîtului şi restului este dat mai jos.

Similar, se obţine şi programul ce determină cîtul şi restul împărţirii polinomului $a_1x^n+a_2x^{n-1}+\ldots+a_{n+1}$ prin polinomul x^2+px+q obţinîndu-se cîtul şi restul $rx+s$.

Programele de împărţire se pot utiliza în descompunerea în factori a polinoamelor în găsirea rădăcinilor unui polinom, în calculul valorii unui polinom.

```
  5 REM  ******************************
 10 REM "*IMPARTIREA UNUI POLINOM DE *
 20 REM "*GRAD N PRINTR-UN POLINOM   *
 30 REM "*DE GRADUL 1  : X+A         *
 40 REM "*****************************
 60 PRINT "N=";
 70 INPUT N
 80 DIM A(N+1),B(N+2)
 90 FOR I = 1 TO N+1
100   PRINT "A(";I;")=";
110   INPUT A(I)
120 NEXT I
130 PRINT "A=";
140 INPUT A
150 B(1)=0
160 FOR I = 2 TO N+2
170   B(I)=A(I-1)-A*B(I-1)
180 NEXT I
185 REM
190 REM "AFISAREA REZULTATULYI
200 REM
210 FOR I=2 TO N+1
220   PRINT "B(";I;")=";B(I)
230 NEXT I
240 PRINT "R=";B(N+2)
250 END
```

```
  5 REM "******************************
 10 REM " IMPARTIREA UNUI POLINOM DE *
 15 REM " GRAD N PRINTR-UN POLINOM   *
 20 REM " DE GRADUL DOI              *
 25 REM "******************************
 30 PRINT "N=";
 40 INPUT N
 45 DIM A(N+1),B(N+3)
 50 PRINT "COEFICIENTII POLINOMULUI "
 60 FOR I=1 TO N+1
 70   PRINT "A(";I;")=";
 80   INPUT A(I)
 90 NEXT I
100 PRINT "COEF. DIVIZORULUI"
110 PRINT "X^2+P*X+Q"
120 PRINT "P=";
130 INPUT P
140 PRINT "Q=";
150 INPUT Q
200 B(1)=0
210 B(2)=0
220 FOR I=3 TO N+3
230   B(I)=A(I-2)-P*B(I-1)-Q*B(I-2)
240 NEXT I
250 REM "AFISAREA REZULTATELOR"
260 FOR I=3 TO N+1
270   PRINT "B(";I;")=";B(I)
280 NEXT I
290 PRINT "AFISARE REST: RX+S "
300 PRINT "R=";B(N+2)
310 PRINT "S=";B(N+3)+P*B(N+2)
320 END
```

12.14. Rezolvarea ecuaţiilor algebrice prin metoda Bairstow

Ecuaţiile algebrice, $f(x)=0$, unde f este un polinom pot fi rezolvate prin metode aproximative. În continuare se va descrie o metodă care constă în descompunerea polinomului în produs de polinoame de gradul doi (metoda

Bairstow). Fie polinomul $f(x)=a_1x^n+a_2x^{n-1}+\ldots+a_{n+1}$ și (p, q) două numere. Împărțind polinomul prin x^2+px+q obținem:

$$\sum_{i=1}^{n+1} a_i x^{n+1-i} = \left(\sum_{i=3}^{n+1} b_i x^{n+1-i}\right)(x^2+px+q)+rx+s$$

Prin identificare se obțin coeficienții b_i, r și s în funcție de p și q: $b_1=b_2=0$ și

(1) $$b_i=a_{i-2}-pb_{i-1}-qb_{i-2}, \quad \text{pentru } 3\leqslant i\leqslant n+3$$

(2) $$\text{iar, } r=b_{n+2} \text{ și } s=b_{n+3}+pb_{n+2}$$

Va trebui să se găsească p și q astfel încît : $r(p, q)=s(p, q)=0$. Acest lucru revine la rezolvarea sistemului :

$$r+\Delta p\frac{\partial r}{\partial p}+\Delta q\frac{\partial r}{\partial q}=0$$

$$s+\Delta p\frac{\partial s}{\partial p}+\Delta q\frac{\partial s}{\partial q}=0$$

Pentru calculul lui $\frac{\partial r}{\partial p}$, se vor calcula $\frac{\partial b_i}{\partial p}$ prin recurență din expresiile obținute prin identificare (1).

$$\frac{\partial b_i}{\partial p}=-b_{i-1}-p\frac{\partial b_{i-1}}{\partial p}-q\frac{\partial b_{i-2}}{\partial p}.$$

Notînd $c_i=\frac{\partial b_i}{\partial p}$ pentru $1\leqslant i\leqslant n+3$, șirul c_i va fi: $c_1=c_2=0$ și $c_i=-b_{i-1}-pc_{i-1}-qc_{i-2}$, pentru $3\leqslant i\leqslant n+3$.

Din formulele (2) se obține :

$$\frac{\partial r}{\partial p}=c_{n+2} \quad \text{și} \quad \frac{\partial s}{\partial p}=-q\,c_{n+1}$$

Pe de altă parte, derivînd formulele (1) se obține :

$$\frac{\partial b_i}{\partial q}=c_{i-1}, \quad \text{deci} \quad \frac{\partial r}{\partial q}=c_{n+1} \quad \text{și} \quad \frac{\partial s}{\partial q}=c_{n+2}+pc_{n+1}.$$

Sistemul (3) se va rezolva prin metoda Cramer :

$$d=c_{n+2}^2+c_{n+1}(c_{n+3}+b_{n+2})$$

$$\Delta p=\frac{c_{n+1}b_{n+3}-b_{n+2}c_{n+2}}{d} \quad \text{și} \quad \Delta q=\frac{-b_{n+1}(qc_{n+1}+pc_{n+2})-b_{n+3}c_{n+2}}{d}.$$

Se defineşte apoi un şir de puncte (p′, q′) care, în principiu, converge către un punct (p, q) încît : x^2+px+q divide pe f (x). Practic, numărul de puncte (iteraţii) este limitat la k=1∅∅, în program, utilizîndu-se ca condiţie de oprire :

$$\frac{|\Delta p|+|\Delta q|}{|p|+|q|}<\varepsilon$$

(ε fiind luat $1\emptyset^{-3}$ în program).

Ecuaţia $x^2+px+q=\emptyset$ este apoi rezolvată în C folosind formulele uzuale. Programul continuă cu polinomul cît $\sum_{i=3}^{n+1} b_i x^{n+1-i}$, dacă gradul său este mai mare decît 2.

```
 60 REM ************************************
 65 REM *" REZOLVAREA ECUATIILOR ALGE-     *
 70 REM *" BRICE-METODA BAIRSTOW-          *
 75 REM ************************************
 80 PRINT "GRADUL N=";
 90 INPUT N
 95 DIM A(N+1),B(N+3),C(N+3)
100 FOR I=1 TO N+1
110    PRINT "A(";I;")=";
120    INPUT A(I)
130 NEXT I
150 P=0
155 Q=0
160 K=100
165 E=.001
170 REM "TEST ASUPRA GRADULUI
180 REM
210 IF N <= 2 THEN 570
220 REM "CAUTARE P SI Q
230 REM
240 REM "INITIALIZARE CONTOR CICLURI
260 J=0
280 IF J>K THEN 990
290 J=J+1
300 REM "CALCULUL COEFICIENTILOR
310 REM "B(I) SI C(I)"
320 B(1)=0
322 B(2)=0
324 C(1)=0
326 C(2)=0
330 FOR I=3 TO N+3
340    B(I)=A(I-2)-P*B(I-1)-Q*B(I-2)
350    C(I)=-B(I-1)-P*C(I-1)-Q*C(I-2)
360 NEXT I
370 REM "CALCUL DP SI DQ"
380 REM
390 X=B(N+2)
392 Y=B(N+3)
394 Z=C(N+1)
396 T=C(N+2)
398 U=C(N+3)
400 D=T^2-Z*(U+X)
410 IF D=0 THEN 990
420 A=(Z*Y-X*T)/D
430 B=(-X*(Q*Z+P*T)-Y*T)/D
440 REM "NOILE P SI Q"
450 P=P+A
455 Q=Q+B
460 F=(ABS(A)+ABS(B))/(ABS(P)+ABS(Q))
470 IF F>E THEN 280
480 REM "S-A GASIT POLIN. FACTOR"
485 REM
490 GOSUB 670
500 REM "INLOCUIREA POLINOMULUI "
510 REM
520 N=N-2
530 FOR I=1 TO N+1
540    A(I)=B(I+2)
550 NEXT I
560 GOTO 170
570 REM "ULTIMUL FACTOR"
580 REM
590 IF N=2 THEN 630
600 X=-A(2)/A(1)
605 Y=0
610 GOSUB 800
620 GOTO 650
630 P=A(2)/A(1)
635 Q=A(3)/A(1)
640 GOSUB 670
650 STOP
670 REM "REZOLVAREA EC.:X^2+P*X+Q=0"
680 REM
690 D=P*P-4*Q
700 IF D<0 THEN 750
710 D=SQR(D)
715 Y=0
720 X=(-P+D)/2
725 GOSUB 800
730 X=(-P-D)/2
735 GOSUB 800
740 RETURN
750 D=SQR(-D)/2
755 X=-P/2
760 Y=D
765 GOSUB 800
770 Y=-D
775 GOSUB 800
780 RETURN
800 REM "AFISAREA REZULTATULUI"
820 IF Y=0 THEN 850
830 PRINT X;"+(";Y;")I"
840 RETURN
850 PRINT X
970 RETURN
990 PRINT "PROCES DIVERGENT"
999 END
```

12.15 Metoda celor mai mici pătrate

Metoda celor mai mici pătrate este o tehnică curentă de determinare a unei curbe $y=f(x)$ ce aproximează o mulţime de puncte date:

$$(Y_1, X_1), (Y_2, X_2), \ldots, (Y_M, X_M).$$

Metoda constă în minimizarea sumei patratelor „distanţelor": $d_1^2+d_2^2+\ldots+d_M^2$, unde $d_i=y_i-f(x_i)$ este distanţa între ordonata punctului dat şi cea aproximată. Metoda este aplicată frecvent pentru funcţii exponenţiale sau polinomiale. În toate cazurile, metoda necesită rezolvarea unor sisteme de ecuaţii algebrice liniare, necunoscutele fiind coeficienţii ecuaţiei curbei. De exemplu, se doreşte aproximarea a M puncte date prin curba $y=ax^b$. Pentru aceasta trebuie rezolvat un sistem de două ecuaţii cu necunoscutele a şi b. Ecuaţiile se obţin prin logaritmarea ecuaţiei $y=ax^b$ şi anularea derivatelor parţiale în raport cu log. a şi b:

$$M\cdot\log a+\sum_{i=1}^{M}(\log x_i)\, b=\sum_{i=1}^{M}\log y_i$$

$$\left(\sum_{i=1}^{M}\log x_i\right)\log a+\left(\sum_{i=1}^{M}(\log x_i)^2\right) b=\sum_{i=1}^{M}(\log x_i)(\log y_i)$$

Aceste ecuaţii sînt liniare în log a şi b.

Dacă se doreşte aproximarea prin curba $y=ae^{bx}$, atunci trebuie să se rezolve sistemul:

$$M\cdot\log a+\left(\sum_{i=1}^{M}x_i\right) b=\sum_{i=1}^{M}\log y_i$$

$$\left(\sum_{i=1}^{M}x_i\right)\log a+\left(\sum_{i=1}^{M}x_i^2\right) b=\sum_{i=1}^{M}x_i\log y_i.$$

Pentru aproximarea printr-o funcţie polinom: $Y=C_1+C_2X+\ldots+C_{n+1}X^n$, coeficienţii C_i se vor obţine prin rezolvarea sistemului:

$$M\cdot C_1+\left(\sum_{i=1}^{M}x_i\right)C_2+\ldots+\left(\sum_{i=1}^{M}x_i^n\right)C_{n+1}=\sum_{i=1}^{M}y_i$$

$$\left(\sum_{i=1}^{M}x_i\right)C_1+\left(\sum_{i=1}^{M}x_i^2\right)C_2+\ldots+\left(\sum_{i=1}^{M}x_i^{n+1}\right)C_{n+1}=\sum_{i=1}^{M}x_i\, y_i$$

$$\left(\sum_{i=1}^{M}x_i^n\right)C_1+\left(\sum_{i=1}^{M}x_i^{n+1}\right)C_2+\ldots+\left(\sum_{i=1}^{M}x^{2n}\right)C_{n+1}=\sum_{i=1}^{M}x_i^n\, y_i.$$

Toate aceste exemple conduc la rezolvarea unor sisteme de ecuații liniare pentru care vom utiliza instrucțiunile matriceale disponibile în limbajul BASIC.

Exemplu de program

Se dorește găsirea unei funcții (putere, exponențială, sau un polinom de grad mai mic decît 5), care aproximează cel mai bine un număr de puncte date (maximum 100). Coordonatele x_i, y_i ale punctelor vor fi convertite în logaritmii lor de către program, dacă va fi necesar. Numărul de puncte date și tipul curbei de aproximare fiind variabile, se vor utiliza facilitățile de redimensionare a tablourilor.

Programul va imprima ecuația curbei de aproximare cu coeficienții calculați, lista coordonatelor punctelor date și lista valorilor funcției ($y(x_i)$) ce le aproximează, precum și eroarea comisă (suma pătratelor). Valoarea erorii va fi utilizată pentru alegerea tipului funcției de aproximare. Variabilele utilizate de program sînt următoarele:

X= vector de 100 de elemente ce va conține valorile x_i introduse

Y= vector de 100 de elemente care va conține valorile y_i introduse

A= matrice de 10 linii și 10 coloane ce va conține coeficienții necunoscutelor sistemului de ecuații liniare

B= inversa matricei A

C= vector de 10 elemente conținînd necunoscutele sistemului.

D= vector de 10 elemente ce conține al doilea membru al sistemului

M= numărul de puncte

N= variabilă ce indică metoda de aproximare dorită:

N=0 pentru funcție putere $y=ax^b$

N=1 pentru funcție exponențială $y=ae^{bx}$

N=2, 3 ..., 10 pentru funcție polinom $y=\sum_{i=1}^{N} C_i x^{i-1}$

N1= numărul de ecuații simultane

N1=2 dacă N=0 sau 1

N1=N dacă N=2, 3, ..., 10

Programul funcționează astfel

1. Citește datele
   a) Citește M valori pentru y_i și M valori pentru x_i
   b) Citește valoarea lui N care dă tipul curbei de utilizat
2. Calculează $\log x_i$ și $\log y_i$, dacă N=0, sau $\log y_i$ dacă N=1, pentru i=1, 2, ..., M.
3. Calculează elementele tablourilor A și D, utilizînd formulele corespunzătoare curbei alese
4. Rezolvă sistemul de ecuații.
5. Afișează ecuația curbei
6. Calculează valorile $y(x_i)$ pentru i=1, 2, ..., M
7. Calculul erorii, suma pătratelor distanțelor $d_1^2+d_2^2+\ldots+d_M^2$
8. Afișarea coordonatelor x_i, y_i și a valorii funcției $y(x_i)$, pentru i=1, 2, ..., M, apoi afișarea erorii.

De notat că dacă N=0 sau 1, trebuie reconvertite valorile $\log y_i$ și $\log x_i$ în y_i și respectiv în x_i.

9. Se reprezintă grafic punctele (x_i, y_i) și apoi se trasează curba care unește punctele de interpolare calculate.
10. Dacă se dorește o nouă rulare cu aceleași date, se revine la punctul 1.b).

Instrucțiunile MAT ZER din liniile 2Ø0, 2Ø5 realizează redimensionarea matricei A și a vectorului D la fiecare nouă execuție a programului. În liniile 37Ø și 375 tablourile B și C sînt implicit redimensionate pentru a corespunde matricei A și vectorului D. Rezolvarea sistemului de ecuații este substanțial simplificată prin utilizarea instrucțiunilor matriceale (liniile 37Ø și 375).

```
1 REM "***************************"
2 REM "* APROXIMARE PRIN METODA *"
3 REM "* CELOR MAI MICI PATRATE *"
4 REM "***************************"
5 DIM X(100),Y(100),Z(100)
6 REM "CITIREA DATELOR"
7 PRINT "CITE PUNCTE DEFINITI ?"
8 PRINT "M="
9 INPUT M
10 PRINT "INTRRODUCETI ORDONATELE Y"
11 MAT INPUT Y(M)
12 PRINT "INTRODUCETI ABSCISELE X"
13 MAT INPUT X(M)
14 REM "CITIREA DATELOR"
15 PRINT "INTRODUCETI:N=0 PENTRU PUTERE"
16 PRINT "              N=1 PENTRU EXPONENTIALA"
17 PRINT "              N=NR. TERMENI POLINOM"
18 PRINT "N="
19 INPUT N
100 REM "CALCULUL LOG(X) SI LOG(Y)"
120 REM "DACA E NECESAR"
130 IF N>=2 THEN 170
135   FOR I=1 TO M
140     Y(I)=LOG(Y(I))
145   NEXT I
150   IF N=1 THEN 170
155     FOR I=1 TO M
160       X(I)=LOG(X(I))
165     NEXT I
170 REM "CALCULUL MATRICEI A"
175 REM "SI A VECTORULUI D"
185 N1=N
190 IF N1>=2 THEN 200
195   N1=2
200 MAT A=ZER(N1,N1)
205 MAT D=ZER(N1)
210 A(1,1)=M
211 N2=2*N1-1
215 FOR D1=2 TO N2
220    S=0
225    FOR K=1 TO M
230       S=S+X(K)^(D1-1)
235    NEXT K
240    FOR J=1 TO D1
245       S2=D1-J+1
250       IF J>N1 THEN 265
255          IF S2>N1 THEN 265
260             A(S2,J)=S
265    NEXT J
270    IF D1>N1 THEN 285
275      FOR K=1 TO M
280         D(D1)=D(D1)+Y(K)*X(K)^(D1-1)
281      NEXT K
285 NEXT D1
286 FOR K=1 TO M
287   D(1)=D(1)+Y(K)
288 NEXT K
290 REM
360 REM "REZOLVAREA SISTEMULUI"
365 REM
```

```
370 MAT B=INV(A)
375 MAT C=B*D
380 REM
385 REM "TIPARIREA ECUATIEI"
390 REM
395 IF N>1 THEN 430
400   C1=EXP(C(1))
405   IF N=1 THEN 420
410     PRINT "PUTEREA Y=";C1;"*X^";C(2)
415     GOTO 490
420   PRINT "EXPONENTIALA Y=";C1;"*EXP(";C(2);"*X)"
425   GOTO 490
430 IF C(2)>=0 THEN 445
435   PRINT "POLINOM Y=";C(1);C(2);"*X";
440   GOTO 450
445 PRINT "POLINOM Y=";C(1);"+";C(2);"*X";
450 IF N=2 THEN 485
455   FOR I=3 TO N
460     IF C(I)>=0 THEN 475
465       PRINT C(I);"*X^";I-1;
470       GOTO 480
475      PRINT "+";C(I);"*X^";I-1;
480   NEXT I
485 PRINT
490 REM
495 REM "AFISAREA REZULTATELOR"
500 REM
505 IF N>=2 THEN 545
510   FOR I=1 TO M
515     Y(I)=EXP(Y(I))
520   NEXT I
525   IF N=1 THEN 545
530     FOR I=1 TO M
535       X(I)=EXP(X(I))
540     NEXT I
545 PRINT
550 PRINT "Y(DAT)","Y(CALCULAT)"
555 S=0
560 FOR I=1 TO M
565   IF N>=2 THEN 595
570     IF N=1 THEN 585
575     Y1=C1*X(I)^C(2)
580     GOTO 615
585   Y1=C1*EXP(C(2)*X(I))
590     GOTO 615
595   Y1=C(1)
600   FOR J=2 TO N
605     Y1=Y1+C(J)*X(I)^(J-1)
610   NEXT J
615   S=S+(Y(I)-Y1)^2
620   PRINT Y(I),Y1
621   Z(I)=Y1
625 NEXT I
630 PRINT
635 PRINT "EROAREA=";S
640 REM "CALCULUL VALORILOR MAXIME"
641 REM "PENTTRU CELE DOUA COORDONATE"
645 U1=ABS(X(1))
650 U2=ABS(Y(1))
655 FOR I=2 TO M
660   IF U1>=ABS(X(I)) THEN 670
665     U1=ABS(X(I))
670   IF U2>=ABS(Y(I)) THEN 680
675     U2=ABS(Y(I))
680 NEXT I
685 WINDOW -U1,U1,-U2,U2
690 T=U2/50
695 INIT
700 FOR I=1 TO M
705   MOVE X(I),Y(I)-T
710     DRAW X(I),Y(I)
720 NEXT I
725 MOVE X(1),Z(1)
730 FOR I=2 TO M
735   DRAW X(I),Z(I)
740 NEXT I
745 PRINT "INTRODUCETI VALOAREA 1 PENTRU"
746 PRINT "OPRIREA EXECUTIEI"
747 INPUT S
748 IF S<>1 THEN 15
750 END
```

10.10. Transformata Fourier rapidă

Transformata Fourier constituie de mai mulţi ani un instrument matematic utilizat în numeroase domenii, ca de exemplu în optică, acustică, fizică cuantică, telecomunicaţii, teoria sistemelor şi a proceselor aleatoare etc.

Fourier a demonstrat că orice semnal periodic poate fi considerat ca o combinaţie de oscilaţii sinusoidale de frecvenţe fØ, 2fØ, 3fØ ... Astfel, dacă x (t) reprezintă amplitudinea semnalului în funcţie de timp, x (t) se poate scrie ca suma unui număr oarecare de funcţii sinusoidale. Deoarece pentru fiecare semnal, fazele iniţiale nu sînt identice, sin ωt devine sin (ωt+φ):

$$\sin(\omega t+\varphi)=\sin \omega t \cos \varphi+\sin \varphi \cos \omega t$$

unde cos φ şi sin φ pot fi consideraţi coeficienţi.

Astfel, toate funcţiile periodice se pot descrie în felul următor:

$$\begin{aligned} x(t) = & A_0+a_1 \sin \omega t+b_1 \cos \omega t \\ & +a_2 \sin 2\omega t+b_2 \cos 2\omega t \\ & +a_3 \sin 3\omega t+b_3 \cos 3\omega t \\ & +\ldots \end{aligned}$$

unde: $\omega=\frac{2\pi}{T}=2\pi f$ (T este perioada şi f frecvenţa semnalului);

$a_1, a_2, \ldots, b_1, b_2, \ldots$ sînt constante care reprezintă amplitudinea fiecăreia din componente;

A_0 este valoarea medie.

Această relaţie reprezintă descompunerea în serie Fourier a funcţiei periodice x (t). În continuare, problema principală este calcularea coeficienţilor $A_0, a_1, b_1, a_2, b_2, \ldots$, care se numesc coeficienţii Fourier.

Deoarece AØ este valoarea medie a semnalului periodic x (t), acest coeficient se calculează imediat cu relaţia:

$$A_0=\frac{1}{T}\int_{\emptyset}^{T} x(t)\,dt$$

Fourier a demonstrat că ceilalţi coeficienţi se obţin din formulele:

$$a_n=\frac{2}{T}\int_{\emptyset}^{T} x(t) \sin n\omega t dt$$

$$b_n=\frac{2}{T}\int_{\emptyset}^{T} x(t) \cos n\omega t dt.$$

Se defineşte transformata Fourier a unui semnal continuu x (t) prin integrala:

$$X(f)=\int_{-\infty}^{+\infty} x(t)\, e^{-2i\pi f t}dt$$

Parametrii t şi f reprezintă în general timpul şi frecvenţa, dar foarte bine acest calcul se poate aplica la o mare varietate de fenomene unde t şi f pot reprezenta alţi parametri.

Transformata Fourier inversă se defineşte prin relaţia:

$$x(t)=\int_{-\infty}^{+\infty} X(f)\, e^{2i\pi ft}\, df$$

Prin utilizarea dispozitivelor numerice se obţine adesea un semnal discret x (n), printr-o eşantionare a semnalului iniţial x (t). În acest caz integrala Fourier definită mai sus poate fi înlocuită prin transformata Fourier discretă:

$$X(k)=\sum_{n=\emptyset}^{N-1} x(n)\, e^{-2i\pi nk/N}$$

unde k = 0, 1, ..., N—1.

Mulţimea celor N valori x (n) constituie o reprezentare discretă a funcţiei x (t), iar cele N valori X (k) o reprezentare a spectrului X (f):

$$x(n) \Leftrightarrow x(t)$$
$$X(k) \Leftrightarrow X(f)$$

Algoritmul transformatei Fourier rapide permite reducerea timpului de execuţie a unei transformate Fourier discrete.

Transformata Fourier discretă a unei secvenţe finite de valor {x (n)}, $\emptyset \leqslant n \leqslant N-1$ poate fi prezentată într-o manieră mai practică:

$$X(k)=\sum_{n=\emptyset}^{N-1} x(n)\cdot w^{nk}$$

unde

$$W^{nK}=e^{-2i\pi nk/N}$$

Această relaţie arată că pentru secvenţă de N numere o evaluare directă necesită $(N-1)^2$ înmulţiri şi N.(N—1) adunări. Pentru valori ale lui N, de exemplu 1∅∅∅, ca ordin de mărime, un calcul direct implică o cantitate foarte mare de calcule, chiar şi pentru un sistem puternic.

Principiul transformatei Fourier rapide constă în separarea secvenţei iniţiale de N valori în două secvenţe mai scurte ale căror transformate discrete pot fi combinate pentru a produce transformata discretă a secvenţei de N puncte.

Astfel, dacă N este par şi secvenţa originală a fost împărţită în două secvenţe de N/2 puncte, este necesar un număr de înmulţiri de ordinul $N^2/2$ pentru a evalua transformata discretă pe N puncte. Dacă N/2 este la rîndul său par, atunci secvenţa de N/2 valori se poate de asemenea împărţi în două secvenţe de cîte N/4 valori fiecare, pentru care se calculează independent transformatele şi procedeul poate continua, obţinîndu-se de fiecare dată creşterea vitezei cu un factor de aproximativ 2. Dacă N este o putere a lui 2, procesul se poate repeta pînă la calcularea în final a transformatelor pentru două puncte.

Programul BASIC prezentat efectuează calculul transformatei Fourier discrete pentru un semnal definit, introdus în liniile 170 la 200:

$$x(n) = \sin(2\pi n \cdot 8{,}555/N)$$

unde $n = 1, 2, \ldots N$

```
10  REM"TRANSFORMATA FOURIER RAPIDA"
40  INPUT N
45  PRINT"NUMAR DE VALORI DISCRETE : ";N
50  M-LOG(N)/LOG(2)
60  DIM P(M+1)
70  DIM R(N)
80  DIM Q(N)
90  DIM X(N)
100 P(1)-2
105 P1=4*ATN(1)
110 FOR K-1 TO M
120   P(K+1)=2*P(K)
130 NEXT K
140 A-2*PI/N
150 N0-8.555
160 B-N0*A
170 FOR K-1 TO N
180   R(K)-SIN(B*K)
190   Q(K)-0
200 NEXT K
210 M2-M/2
241 M9-N-1
242 J-1
250 FOR I-1 TO M9
260   IF I>-J THEN 300
270   T1=R(J)
271   T2-Q(J)
280   R(J)-R(I)
281   Q(J)-Q(I)
290   R(I)-T1
291   Q(I)-T2
300   K=M2
310   IF K>-J THEN 320
311   J-J-K
312   K-K/2
313   GOTO 310
320   J-J+K
330 NEXT I
370 FOR K-1 TO M
380   L9-P(K)
381   L1-L9/2
390   U9-1
391   U8-0
400   W9-COS(P1/L1)
401   W8-SIN(P1/L1)
410   FOR J-1 TO L1
420     FOR I=J TO M STEP L9
430       B9=R(I)
431       B8=Q(I)
440       I9=I+L1
450       T1=R(I9)*U9-Q(I9)*U8
460       T2=R(I9)*U8+Q(I9)*U9
470       R(I9)=B9-T1
471       Q(I9)=B8-T2
480       R(I)-B9+T1
481       Q(I)=B8+T2
490     NEXT I
500     U7=U9*W9-U8*W8
510     U8=U9*W8+U8*W9
520     U9=U7
530   NEXT J
540 NEXT K
580 FOR K-1 TO N
590   X(K)-((R(K)*R(K))+(Q(K)*Q(K)))
600 NEXT K
610 PRINT"K";"  "
620 PRINT" K    REAL        IMAGINAR        AMPLITUDINE"
625 PRINT"------------------------------------------------"
630 FOR K-1 TO N
640   PRINT K;"  ";
650   PRINT INT(10000*R(K)),INT(10000*Q(K)),INT(10000*X(K))
655   PRINT
660 NEXT K
900 END
```

Pornind de la secvenţa de N valori x (n), se generează două secvenţe de N/2 puncte fiecare x_1 (n) şi x_2 (n), prima conţinînd termenii de rang par din secvenţa iniţială, iar a doua termenii de rang impar:

$$x_1(n) = x(2n)$$

$$x_2(n) = x(2n+1)$$

unde n=0, 1, 2, ... N/2—1

Se poate arăta că transformata Fourier discretă a secvenţei iniţiale se scrie făcînd compunerea transformatelor Fourier pentru cele două secvenţe iniţiale x_1 (n) şi x_2 (n):

$$X(k) = X_1(k) + W_N^k X_2(k)$$

unde X (k), X_1 (k) şi X_2 (k) sînt transformatele Fourier discrete ale secvenţelor x (n), x_1 (n) şi x_2 (n) şi:

$$W_N^k = e^{-i(2\pi/N)k}$$

Datorită proprietăţilor de periodicitate ale transformatei Fourier, relaţia precedentă se poate scrie de asemena:

$$X(k) = X_1(k-N/2) - W_N^k X_2(k-N/2)$$

pentru k cuprins între N/2 şi N—1, iar:

$$W_N^{k+N/2} = W_N^k \cdot W_N^{N/2} = -W_N^k$$

12.17. Simularea salturilor unei mingi

Se va descrie mişcarea unei mingi de cauciuc care sare sus-jos sub efectul greutăţii proprii şi în acelaşi timp se deplasează pe orizontală cu viteză constantă. Se presupun date: înălţimea iniţială de la sol (H), viteza orizontală (V) şi numărul de salturi ale mingii (N). Se dă de asemenea coeficientul de restituire (C) definit ca raportul vitezelor imediat după şi înainte de salt.

Pentru a calcula poziţia mingii în timp se dă un increment mic de timp (D) şi se aplică legile fizicii pe acest interval:

T (I+1) = T (I)+D

X (I+1) = X (I)+V*D

Z (I+1) = Z (I)—G*D

Y (I+1) = Y (I)+Ø.5*(Z (I)+Z (I+1))*D

Unde:

X este deplasarea pe orizontală (iniţial nulă), Z este viteza pe verticală (de asemenea nulă la plecare), Y este înălţimea deasupra solului, G este accele-

rația gravitațională (9,81 m/s^2). Indicii I și I+1 corespund valorilor diferitelor variabile la începutul și la sfîrșitul incrementului de timp.

Dacă are loc o ciocnire pe durata incrementului de timp, formulele ce se aplică sînt puțin modificate. Prezența unei ciocniri este semnalată printr-o valoarea negativă a lui Y (I+1), ce ar fi imposibilă fizic. Cînd se produce aceasta, Z (I+1) și (I+1) se recalculează ca mai jos.

Mai întîi se calculează timpul necesar mingii pentru a atinge pămîntul, plecînd din poziția sa de la începutul incrementului de timp. Dacă se notează acest timp D1, atunci:

D1=D*Y (I)/(Y(I)—Y(I+1))

Se calculează apoi viteza pe verticală imediat înainte de ciocnire:

Z=Z (I)—G*D1

Viteza pe verticală imediat după ciocnire va fi deci:

Z1=—C*(Z (I)—G*D1)

Înălțimea deasupra solului la sfîrșitul incrementului de timp este:

Y (I+1)=∅.5*(Z1+Z(I+1))*(D—D1)

iar viteza pe verticală la sfîrșitul incrementului de timp:

Z (I+1)=Z1—G* (D—D1)

Programul funcționează astfel:

1. Citește valorile H, V, N, C și D și testează validitatea lor

2. Inițializează parametrii:

I=1 (contor de incrementare) X (1)=0

B=0 (contor de salturi) Z (1)=∅

T (1)=∅ Y(1)=H

3. Calculează deplasarea pe orizontală și pe verticală, și viteza pe verticală cu ajutorul formulelor de mai sus.

4. Dacă mingia lovește pămîntul pe parcursul incrementului de timp se testează dacă trebuie calculat saltul următor sau se termină programul.

a) Dacă B<N se recalculează viteza pe verticală și deplasarea pe verticală cu formulele modificate, se incrementează contorul de salturi (B=B+1) și se trece la incrementul de timp următor.

b) Dacă B=N se calculează timpul și deplasarea pe orizontală în momentul ciocnirii.

5. Afișează valorile lui X și T urmate de o tabelare completă pentru T, X, Y și Z.

6. Trasează grafic Y funcție de T.

7. Se reîntoarce la pasul 1. Dacă se introduce H=∅ se iese din program.

Programul acceptă 1∅∅ de incrementări de timp. D trebuie ales astfel încît să existe între 8 şi 2∅ de puncte de fiecare salt.

Calculul valorilor T (I+1), X (I+1), Z (I+1) şi Y (I+1) este efectuat de un subprogram.

Se observă că de fapt programul integrează ecuaţia diferenţială $d^2y/dt^2 = -g$ cu ajutorul metodei Euler modificată.

```
002 REM "**********************************"
005 REM "* SIMULAREA SARITURII UNEI MINGI *"
010 REM "**********************************"
015
020 REM "INTRODUCEREA DATELOR SI VERIFICAREA LOR"
025
030 DIM X(100),Y(100),Z(100),T(100),D$(1)
035 PRINT "INALTIME INITIALA(M);(0=STOP)";
040 INPUT H
045 IF H>=0 THEN 060
050 PRINT "INALTIME NEGATIVA-EROARE!"
055 GO TO 005
060 IF H=0 THEN 600
065 PRINT "VITEZA ORIZONTALA(M/S)";
070 INPUT V
075 PRINT "NUMAR DE SALTURI";
080 INPUT N
085 IF N>=0 THEN 100
090 PRINT "NUMAR DE SALTURI NEGATIV-EROARE!"
095 GO TO 075
100 PRINT "COEFICIENT DE RESTITUIRE";
105 INPUT C
110 IF C<=1 THEN 125
115 PRINT "COEFICIENT SUPRAUNITAR-EROARE!"
120 GO TO 100
125 PRINT "INCREMENT DE TIMP(S)";
130 INPUT D
135 IF D>0 THEN 150
140 PRINT "INCREMENT NEGATIV SAU NUL-EROARE!"
145 GO TO 125
150 PRINT
155 REM "INITIALIZAREA PARAMETRILOR"
160
165 B=T(1)=X(1)=Z(1)=0
170 Y(1)=H
175 G=9.81
180
185 REM "CALCULUL VITEZEI SI AL DEPLASARII"
190 REM "PENTRU FIECARE INCREMENT"
195
200 FOR I=1 TO 99
205    GOSUB 550
210    IF Y(I+1)>0 THEN 280
215    IF B=N THEN 310
225    D1=D*Y(I)/(Y(I)-Y(I+1))
230    D2=0
235    Z1=-C*(Z(I)-G*D1)
240    Z(I+1)=Z1-G*(D-D1-D2)
245    Y(I+1)=0.5*(Z1+Z(I+1))*(D-D1-D2)
250    B=B+1
255    IF Y(I+1)>0 THEN 280
260    IF B=N THEN 310
265    D2=D2+2*Z1/G
270    Z1=C*Z1
275    GO TO 240
280 NEXT I
282 I=99
285 GO TO 320
290
295 REM "ULTIMUL SALT"
300
310 D=D*Y(I)/(Y(I)-Y(I+1))
315 GO SUB 550
320 I1=I+1
325 T1=T(I1)
330 X1=X(I1)
335
340 REM "AFISAREA REZULTATELOR NUMERICE"
345
350 PRINT "DISTANTA ORIZONTALA PARCURSA=";X1;" M"
355 PRINT "TIMPUL CERUT=";T1;" S"
360 PRINT
361 PRINT "TIMP","DIST.ORIZ.","INALTIME","VIT.VERT."
362 PRINT "================================"
365 FOR I=1 TO I1
370    PRINT "T=";T(I),"X=";X(I),"Y=";Y(I),"Z=";Z(I)
371    PRINT
375 NEXT I
380 PRINT
385
390 PRINT "SOLUTIA GRAFICA A PROBLEMEI(D/N)";
395
400 INPUT D$
405 IF D$<>"D" THEN 035
410 INIT
415 VIEWPORT 20,140,10,90
420 L=-I1/10
425 R=11*I1/10
430 B=-H/10
435 T=11*H/10
440 WINDOW L,R,B,T
441 MOVE L,B
442 DRAW R,B
443 DRAW R,T
444 DRAW L,T
445 DRAW L,B
446
450 REM "TRASAREA AXELOR"
455
460 MOVE L,O
465 DRAW R,O
470 MOVE 1,B
475 DRAW 1,T
480
485 REM "TRASAREA CURBEI"
490
495 MOVE 1,H
500 FOR I=2 TO I1
505    DRAW I,Y(I)
510 NEXT I
515 MOVE L,B
520 GO TO 035
525
530 REM "SUBPROGRAM DE CALCUL AL VITEZEI SI AL
535 REM "DEPLASARII LA SFIRSITUL INCREMENTULUI"
540
550 T(I+1)=T(I)+D
555 X(I+1)=X(I)+V*D
560 Z(I+1)=Z(I)-G*D
565 Y(I+1)=Y(I)+0.5*(Z(I+1)+Z(I))*D
570 RETURN
600 END
```

12.18. Exerciţii de despărţire a cuvintelor în silabe

Programul constituie un exemplu simplu de aplicare a învăţării programate. Se afişează cîte o regulă de despărţire în silabe, urmată de un exemplu. În continuare sînt date mai multe exerciţii de verificare pentru regula respec-

tivă. Regulile și exercițiile sînt codificate prin DATA începînd cu linia 500 din program, putîndu-se adăuga noi reguli și exerciții.

```
10  DIM E$(32)
20  N = 1
30  INIT
40  PRINT " DESPARTIREA CUVINTELOR IN "
45  PRINT " ---------------------------- "
50  PRINT " SILABE "
55  PRINT " ------ "
60  PRINT " REGULA ";N
70  PRINT " -- ----- "
80  GOSUB 340
90  READ E$(TO)
100 PRINT " EXEMPLU ";
110 PRINT E$
120 PRINT
130 PRINT " EXERCITII "
140 READ E$(TO)
150 IF E$(2TO2) = "0" THEN 180
160 PRINT E$
170 GOTO 140
180 READ R
190 I = O
200 PRINT " RASPUNS? "
210 INPUT E
220 IF E = R THEN 260
230 I = 1
240 PRINT " REVEDETI REGULA "
250 GOTO 200
260 IF I <> O THEN 280
270 PRINT " FELICITARI "
280 READ E$(TO)
290 IF E$(2TO2) = "1" THEN 140
300 IF E$(2TO2) = "3" THEN 400
310 N = N+1
320 IF N < 4 THEN 30
330 STOP
340 READ M
350 FOR I = 1 TO M
360 READ F$(TO)
370 PRINT E$
380 NEXT I
390 RETURN
400 READ E$(TO)
410 PRINT E$
420 GOTO 80
500  DATA 3
510  DATA " DACA VOCALA E URMATA DE O "
520  DATA " SINGURA CONSOANA ACEASTA "
530  DATA " TRECE LA SILABA URMATOARE "
540  DATA " LE - GE ; O - RA "
550  DATA " DU - NA - RE = 1 "
560  DATA " DUN - A - RE = 2 "
570  DATA " DUN - AR - E = 3 "
580  DATA " D - UN - A - RE = 4 "
590  DATA "0",1,"1"
600  DATA " HOT - A - RI - RE = 1 "
610  DATA " HO - TAR - IRE = 2 "
620  DATA " HOT - AR -IRE = 3 "
630  DATA " HO - TA - RI - RE = 4 "
640  DATA "0",4,"2"
650  DATA 4
660  DATA " DINTRE DOUA VOCALE SUCCESIVE "
670  DATA " CARE NU FORMEAZA DIFTONG "
680  DATA " PRIMA APARTINE SILABEI DINAINTE "
690  DATA " A DOUA CELEI URMATOARE "
700  DATA " CE - RE - A - LE ; LU - A "
710  DATA " IN - DI - VI - DUA - LI - TA - TE = 1 "
720  DATA " IN - DI - VI - DU -A - LI - TA - TE = 2 "
730  DATA " "0",2,"1"
740  DATA " PO - DE - A - UA = 1 "
750  DATA " PO - DEA - UA = 2 "
760  DATA "0",2,"2"
770  DATA 5
780  DATA " DACA E URMATA DE DOUA "
790  DATA " SAU MAI MULTE CONSOANE "
800  DATA " PRIMA CONSOANA TRECE LA SILABA "
810  DATA " DINAINTE;CEALALTA (CELELALTE) "
820  DATA " LA SILABA URMATOARE "
830  DATA " E - XIS - TA ; CO - REC - TA "
840  DATA " MU - NTE - NI - A = 1 "
850  DATA " MUNT - E - N - IA = 2 "
860  DATA " MUN - TE - N - IA = 3 "
870  DATA " MUN - TE - NI - A = 4 "
880  DATA "0",4,"1"
890  DATA " PER - SON - A - LI - TA - TE = 1 "
900  DATA " PERS - ON - A - LI - TA - TE = 2 "
910  DATA " PER - SON - AL - I - TA - TE = 3 "
920  DATA " PER - SO - NA - LI - TA - TE = 4 "
930  DATA "0",4,"3"
940  DATA " EXISTA EXCEPTII LA ACEASTA REGULA "
950  DATA 3
960  DATA " CIND PRIMA CONSOANA ESTE "
970  DATA " B,C,D,G,P,T,K,F IAR A DOUA L,R "
980  DATA " AMINDOUA TREC LA SILABA URMATOARE
990  DATA " A - BRE - VI - A ; A - CRU "
1000 DATA " COD - RU = 1 "
1010 DATA " CO - DRU = 2 "
1020 DATA "0",2,"1"
1030 DATA " SUP - LU = 1 "
1040 DATA " SU - PLU = 2 "
1050 DATA "0",2,"1"
1060 DATA " DE - MOC - RAT = 1 "
1070 DATA " DE - MO - CRAT = 2 "
1080 DATA "0",2,"1"
1090 DATA " NI - SET - RU = 1 "
1100 DATA " NI - SE - TRU = 2 "
1110 DATA "0",2,"2"
2000 END
```

12.19. Verificarea cunoștințelor de geografie

Exemplul ales în cadrul programului este axat pe verificare a cunoașterii reședințelor de județ. Examinatului i se solicită să răspundă la 10 întrebări, afișînd un număr de 10 județe alese aleator și cerînd să se introducă numele reședinței județului respectiv. După parcurgerea celor 10 întrebări, se afișează numărul de răspunsuri corecte. Dacă se răspunde greșit la o întrebare, se indică și răspunsul corect.

Programul poate fi extins fără dificultăți pentru orice gen de teste asemănătoare (verificarea cunoașterii regulilor de circulație, formule, definiții etc.)

```
10  PRINT " VERIFICAREA CUNOSTINTELOR "
15  PRINT " LA GEOGRAFIE "
20  PRINT " DENUMIRE JUDET  - RESEDINTA "
25  DIM-M$(50),S$(50),V$(50).
30  N = O
35  T = 39
40  FOR I = 1 TO 10
45  FOR J = 1 TO INT(RND(X)*T+1)
50  READ M$(TO),S$(TO)
55  NEXT J
60  PRINT " DATI RESEDINTA JUDETULUI ";M$
65  INPUT V$(TO)
70  IF V$ <> S$ THEN 85
75  H = H+1
80  GOTO 90
85  PRINT " RASPUNS CORECT AR FI ";S$
90  RESTORE
95  NEXT I
100 PRINT " DIN 10 INTREBARI ";H;" ATI STIUT
110 STOP
120 DATA " JUDET  1 ";" RESEDINTA 1 "
130 DATA .
400 DATA ' JUDET N ' ; ' RESEDINTA N
410 END
```

12.20. Verificarea cunoştinţelor unui grup de candidaţi

Iniţial, programul solicită prin dialog crearea tabelului cu întrebări. Fiecărei întrebări, operatorul îi asociază 3 răspunsuri posibile cu indicarea răspunsului corect. Se cere apoi numărul total de persoane de examinat. Pentru

```
2 DIM B(20)
3 DIM U(100,50),T(50),S(50)
4 MAT B = ZER.
5 INIT
6 PRINT " INTRODUCETI INTREBARILE? (DA/NU) "
7 INPUT A$
8 IF A$ = "DA" THEN 1000
10 PRINT " DATI NUMERE DE RASPUNSURI CORECTE? (DA/NU) "
11 INPUT A$
12 IF A$ = "DA" THEN 1170
15 IF B(1) <> O THEN 20
16 PRINT " FARA RASPUNSURI INTRODUSE "
18 STOP
20  PRINT " VERIFICARE DE CUNOSTIMTE
30  PRINT " CITE PERSOANE EXAMINATI
40  INPUT M
50  MAT U = ZER
60  MAT T = ZER
70  REM " INCEPE VERIFICAREA "
80  FOR I = 1 TO M
90  FOR J = O TO N-1
95 IF W = O THEN 140
100 PRINT G$(J+1)
110 PRINT R$(J*3+1)
120 PRINT R$(J*3+2)
130 PRINT R$(J*3+3)
140 PRINT " DATI RASPUNSUL (1,2,3)
150 INPUT R
160 IF B(J+1) = R THEN 210
170 PRINT " RASPUNS ERONAT "
180 PRINT " RASPUNS CORECT ESTE ";
185 IF W = O THEN 195
190 PRINTR$(J*3+B(J+1)(3TO))
192 GOTO 230
195 PRINT B(J+1)
200 GOTO 230
210 PRINT " RASPUNS CORECT "
220 U(I,J+1) = 1
230 NEXT J
240 Q = O
250 FOR K = 1 TO N
260 Q = Q+U(I,K)
270 NEXT K
280 T(I) = Q
290 PRINT " DIN ";N;" INTREBARI CUNOASTE ",
295 PRINT Q
300 NEXT I
310 J = O
320 FOR I = 1 TO M
330 J = J+T(I)
350 NEXT I
360 K = J/M
370 PRINT " IN MEDIE CUNOSC ";K;" INTREBARI "
```

```
380 FOR I = 1 TO M
390 S(I) = (T(I)*100)/N
400 NEXT I
405 REM " GRAFICUL; X=NR. INTREBARE, Y=CITI STIU "
410 INITP
420 WINDOW -10,N+10,-10,110
430 MOVE -10,0
440 DRAW N+10,0
450 MOVE 0,-10
460 DRAW 0,110
470 MOVE -1,25
480 DRAW 1,25
490 MOVE -2,50
500 DRAW 2,50
510 MOVE -1,75
520 DRAW 1,75
530 MOVE -2,100
540 DRAW 2,100
550 FOR I = 1 TO N
560 MOVE I,0
570 DRAW I,S(I)
580 NEXT I
590 STOP
1000 REM " INITIALIZARE "
1005 W = 1
1010 DIM G$(20,50),R$(3*20,30),B(20)
1020 PRINT " CITE INTREBARI INTRODUCETI? "
1030 INPUT N
1050 PRINT " DATI INTREBAREA SI CELE "
1055 PRINT " TREI RASPUNSURI "
1060 FOR I = 0 TO N-1
1070 PRINT " INTREBAREA ";I+1
1080 INPUT G$(I+1)
1090 FOR J = 1 TO 3
1100 PRINT " RASPUNSUL ";J
1110 INPUT R$(I*3+J)
1120 NEXT J
1130 PRINT " DATI NUMARUL RASPUNSULUI "
1135 PRINT " CORECT "
1140 INPUT B(I+1)
1150 NEXT I
1160 GOTO 20
1170 PRINT " CITE INTREBARI AVETI? "
1172 INPUT N
1175 FOR I = 0 TO N-1
1176 PRINT " RASPUNSUL LA INTREBAREA ";I+1
1180 INPUT B(I+1)
1190 NEXT I
1200 GOTO 20
```

fiecare persoană în parte, se parcurge tot setul de întrebări, cel examinat indicînd de fiecare dată răspunsul pe care îl consideră ca fiind corect. La răspuns incorect, programul afișează pe cel corect, în vederea însușirii lui. După parcurgerea setului de întrebări, se afișează totalul răspunsurilor corecte date de examinat.

La epuizarea examinării tuturor persoanelor, se afișează numărul mediu de răspunsuri corecte, caracteristic grupului examinat, și se trasează graficul cu numărul de răspunsuri corecte pentru fiecare persoană în parte.

12.21. Ordonarea candidaților după mediile obținute

Exemplul de față realizează ordonarea candidaților după mediile obținute la 5 probe. Se introduce numărul total al candidaților, urmat de numele fiecărui candidat și calificativele obținute la cele 5 probe. Se afișează numele candidaților și media obținută, în ordinea descrescătoare a mediilor.

Prin modificarea liniei 17 se poate modifica numărul de probe.

```
5   PRINT " ORDONAREACANDIDATILOR DUPA MEDIE "
10  PRINT " DATI NUMARUL CANDIDATILOR "
15  INPUT N
17  P = 5
20  DIM A$(N,20),M(N),C(P)
22  MAT M = ZER
25  PRINT " DATI NUMELE SI CALIFICATIVUL "
30  FOR I = 1 TO N
35  INPUT A$(I(TO))
40  M(I) = O
45  MAT INPUT C
50  FOR J = 1 TO P
60  M(I) = M(I)+C(J)
65  NEXT J
70  M(I) = M(I)/P
75  NEXT I
80  I = 1
85  C = M(I)
90  J = I+1
95  K = I
100 IF M(J) <= C THEN 115
105 C = M(J)
110 K = J
115 IF J = N THEN 130
120 J = J+1
125 GOTO 100
130 REM " INVERSAREA ORDINEI "
135 B$ = A$(I(TO))
140 A$(I) = A$(K(TO))
145 A$(K) = B$
150 S = M(I)
155 M(I) = M(K)
160 M(K) = S
165 I = I+1
170 IF I < N THEN 85
175 REM " TIPARIREA "
180 PRINT " NUMELE        "; " MEDIA
185 FOR I = 1 TO N
190 PRINT A$(I),M(I)
195 NEXT I
200 STOP
210 END
```

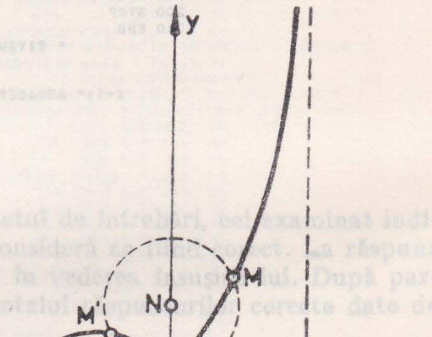
Capitolul 13. Microcalculatorul aMIC în grafică, jocuri, aplicaţii diverse

13.1. Trasarea strofoidei

Programul trasează strofoida, conform ecuaţiei :

$$Y^2 = \frac{X^2 (X + a)}{a - X}$$

Se solicită prin dialog distanţa punctului A faţă de origine. (fig. 13.1)




```
5   PRINT " STROFOIDA "
10  PRINT " DATI VALOAREA A "
15  INPUT A
20  X = A/3
25  Y = SQR(X^2*(X+A)/(A-X))
30  INITP
35  WINDOW -Y,Y,-Y,Y
40  MOVE -Y,O
45  DRAW Y,O
50  MOVE O,-Y
55  DRAW O,Y
60  MOVE X,Y
65  FOR X = A/3 TO -A STEP -1
70  Y = SQR(X^2*(X+A)/(A-X))
75  IF X <= O THEN 90
80  DRAW X,Y
85  GOTO 95
90  DRAW X,-Y
95  NEXT X
100 FOR X = -A TO A/3
105 Y = SQR(X^2*(X+A)/(A-X))
110 IF X >= O THEN 125
115 DRAW X,Y
120 GOTO 130
125 DRAW X,-Y
130 NEXT X
135 STOP
140 END
```

Fig. 13.1. Strofoida.

13.2. Trasarea cicloidei

Programul trasează cicloida, conform ecuației :

$$X = r(t - \sin t)$$

$$Y = r(1 - \cos t)$$

unde : (fig. 13.2)

r este raza cercului,
t este unghiul de rotație,
λ este parametru :

λ este 1 punct pe cerc
λ>1 punct exterior cercului
λ<1 punct interior cercului

Se solicită prin dialog valorile pentru r, λ precum și numărul total de cicluri de trasat.

```
5   PRINT " CICLOIDA "
10  PRINT " DATI RAZA CERCULUI "
15  INPUT R
20  PRINT " DATI PARAMETRUL L "
30  INPUT L
40  PRINT " CITE CICLURI? (1,2,3) "
50  INPUT N
55  IF N <= 3 THEN 65
60  N = 3
65  A = -2*R
70  B = N*2*PI*?
75  INITP
80  WINDOW A,B,A,B
85  REM " TRASAREA AXELOR "
90  MOVE A,O
95  DRAW B,O
100 MOVE O,A
105 DRAW O,B
110 REM " TRASAREA CERCULUI "
115 MOVE R,R
120 FOR J = O TO 2*PI STEP PI/10
125 DRAW R*COS(J),R+R*SIN(J)
130 NEXT J
135 X = O
140 Y = R*(1-L).
145 MOVE X,Y
150 FOR J=O TO N*2*PI STEP PI/10
155 X = R*(J-L*SIN(J))
160 Y = R*(1-L*COS(J))
165 DRAW X,Y
170 NEXT J
175 STOP
180 END
```

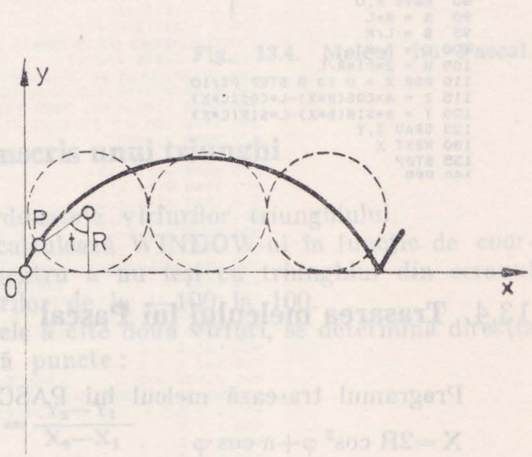


Fig. 13.2. Cicloida.

13.3. Trasarea epicicloidei

Programul trasează epicicloida, după ecuațiile :

$$X = (R+r)\cos\frac{r}{R}t - r\cos\frac{R+r}{R}t$$

$$Y = (R+r)\sin\frac{r}{R}t - r\sin\frac{R+r}{R}t$$

unde : (fig. 13.3)

R — raza cercului fix

r — raza cercului mobil

t — unghiul format de dreapta care unește centrala celor 2 cercuri cu axa x.

Forma curbei depinde de raportul r/R; dacă r=R se obține graficul cardioidei.

Se solicită prin dialog valorile pentru r și R și se trasează epicicloida.

```
5   PRINT " EPICICLOIDA "
10  PRINT " DATI RAZA CERCULUI FIX "
15  INPUT R
20  PRINT " DATI RAZA CERCULUI MOBIL "
25  INPUT L
30  INITP
35  A = R+2*L
40  WINDOW -A,A,-A,A
45  MOVE -A,O
50  DRAW A,O
55  MOVE O,-A
60  DRAW O,A
65  MOVE R,O
70  FOR X = O TO 2*PI STEP PI/10
75  DRAW R*COS(X),R*SIN(X)
80  NEXT X
85  MOVE R,O
90  A = R+L
95  B = L/R
100 C = A/R
105 H = 2*PI*R/L
110 FOR X = O TO H STEP PI/10
115 Z = A*COS(B*X)-L*COS(C*X)
120 Y = A*SIN(B*X)-L*SIN(C*X)
125 DRAW Z,Y
130 NEXT X
135 STOP
140 END
```

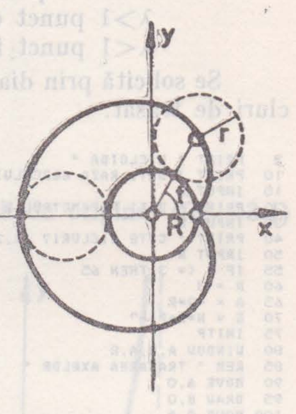


Fig. 13.3. Epicicloidia.

13.4. Trasarea melcului lui Pascal

Programul trasează melcul lui PASCAL pe baza ecuațiilor :

$X=2R\cos^2\varphi+a\cos\varphi$

$Y=2R\cos\varphi\sin\varphi+a\sin\varphi$

unde (fig. 13.4) :

R este raza cercului

φ este unghiul dreptei care se rotește față de axa X

a este parametrul de care depinde forma curbei (dacă a/R=2 se obține graficul cardioidei)

Se solicită prin dialog valorile pentru R și pentru raportul a/R (nu se dă valoarea direct pentru a).

```
5 ' PRINT " MELCUL LUI PASCAL "
10  PRINT " CONC OIDA CERCULUI "
15  PRINT " DATI RAZA CERCULUI "
20  INPUT R
25  PRINT " DATI RAPORTUL A/R "
30  PRINT " FORMA CURBEI DEPINDE DE A/R "
35  PRINT " A/R < 2 CU BUCLA "
40  PRINT " A/R = 2 CARDIOIDA "
45  PRINT " 2 < A/R < 4 FARA BUCLA "
50  PRINT " A/R > 4 CURBA CONVEXA "
55  INPUT N
60  IF N <= 6 THEN 70
65  N = 6
70  A = N*R
75  INITP
80  WINDOW -4*R,2*R+A,-4*R,2*R+A
85  MOVE -4*R,0
90  DRAW 2*R+A,0
95  MOVE 0,-4*R
100 DRAW 0,2*R+A
105 MOVE 2*R,0
110 FOR I = 0 TO 2*PI STEP PI/10
115 DRAW R+R*COS(I),R*SIN(I)
120 NEXT I
122 REM " TRASAREA CURBEI "
125 FOR I = 1 TO 2*PI STEP PI/10
130 Z = COS(I)
135 U = SIN(I)
140 X = 2*R*Z^2+A*Z
145 Y = 2*R*Z*U+A*U
150 DRAW X,Y
155 NEXT I
160 PRINT " MAI VRETI PT. ALT RAPORT?";
165 PRINT " (DA/NU ) "
170 INPUT D$
175 IF D$ = "DA" THEN 25
180 STOP
185 END
```

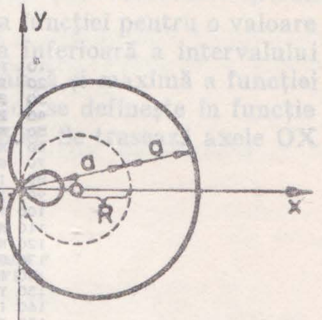


Fig. 13.4. Melcul lui Pascal.

13.5. Trasarea cercului circumscris unui triunghi

Se solicită prin dialog coordonatele vîrfurilor triunghiului.

Întrucît în program nu se calculează WINDOW-ul în funcţie de coordonatele vîrfurilor triunghiului, pentru a nu ieşi cu triunghiul din ecranul TV se recomandă utilizarea valorilor de la —100 la 100.

În program se iau coordonatele a cîte două vîrfuri, se determină direcţia dreptei definite prin aceste două puncte :

$$m=\frac{Y_2-Y_1}{X_2-X_1}$$

Se determină coordonatele punctului median al laturii Prin acest punct trece mediatoarea triunghiului cu direcţia :

$$R=-\frac{1}{m}.$$

Ecuaţia ei (dacă punctul median are coordonatele X, Y) este :

$$y-Y=R(x-X)$$

Dacă intersectăm două mediatoare obţinem coordonatele centrului cercului (B, C) circumscris triunghiului.

Din coordonatele centrului cercului și coordonatele unui vîrf al triunghiului se poate determina raza cercului.

$$R=\sqrt{|B-X_1|^2+|C-Y_1|^2}$$

Se inițializează ecranul și se determină spațiul utilizator: —100, 100— —100, 100. Se trasează triunghiul și cercului circumscris. Se afișează coordonatele centrului cercului și mărimea razei.

```
10  PRINT " CERCUL CIRCUMSCRIS "
20  DIM A(2),B(2),C(2)
30  PRINT " DATI COORDONATELE VIRFURILOR "
40  MAT INPUT A,B,C
50  MAT D = A
60  MAT E = B
70  GOSUB 600
80  IF Y = 0 THEN 400
90  S = R
100 T = B
110 MAT D = C
120 MAT E = B
130 GOSUB 600
140 IF Y = 0 THEN 460
150 Y = (S*B-R*T)/(S-R)
160 IF S > 0 THEN 190
170 X = (Y-B)/R
180 GOTO 200 .
190 X = (Y-T)/S
200 B = ABS(X-A(1))
210 C = ABS(Y-A(2))
220 R = SQR(B^2+C^2)
230 INITP
240 WINDOW -100,100,-100,100
250 MOVE -100,0
260 DRAW 100,0
270 MOVE 0,-100
280 DRAW 0,100
290 MOVE A(1),A(2)
300 DRAW B(1),B(2)
310 DRAW C(1),C(2)
320 DRAW A(1),A(2)
330 MOVE X+R,Y
340 FOR I = 0 TO 2*PI STEP PI/10
350 DRAW X+R*COS(I),Y+R*SIN(I)
360 NEXT I
370 PRINT AT(1,1);"RAZA = ";R
380 PRINT AT(2,1);"X = ";X,"Y = ";Y
390 STOP
400 MAT D = C
410 MAT E = B
420 GOSUB 600
430 IF Y = 0 THEN 510
440 S = R
450 T = B
460 MAT D = A
470 MAT E = C
480 GOSUB 600
490 IF Y = 0 THEN 510
500 GOTO 150
510 PRINT " PUNCTE COLINIARE "
520 GOTO 30
600 Y = 0
610 I = D(2)-E(2)
620 IF I = 0 THEN 730
630 R = -(D(1)-E(1))/I
640 X = E(1)
650 IF E(1) < D(1) THEN 670
660 X = D(1)
670 X = X+ABS((D(1)-E(1))/2)
680 Y = E(2)
690 IF E(2) < D(2) THEN 710
700 Y = D(2)
710 Y = Y+ABS((D(2)-E(2))/2)
720 B = R*(0-X)+Y
730 RETURN
740 END
```

13.6. Graficul funcţiei polinominale

Se introduce gradul polinomului şi valorile tuturor coeficienţilor (zero pentru coeficienţii care lipsesc). Se introduce apoi intervalul de definiţie al funcţiei şi numărului de paşi în care se face trasarea graficului. Un număr mai mare de paşi face trasarea mai exactă, dar mai lentă. Se va corela acest număr cu mărimea intervalului de definiţie.

Subprogramul de la linia 500 calculează valoarea funcţiei pentru o valoare dată a variabilei. La linia 170 se apelează cu limita inferioară a intervalului de definiţie. În bucla 200—280 se reţine valoarea minimă şi maximă a funcţiei (în P şi R) pe intervalul de definiţie. Spaţiul utilizator se defineşte în funcţie de limitele de definiţie şi de valorile P şi R în linia 330. Se trasează axele OX şi OY şi forma funcţiei pe acest interval.

```
10  PRINT " GRAFICUL FUNCTIEI POLINOMIALE "
15  PRINT " DAT PRIN GRAD SI COEFICIENTI "
20  PRINT " DATI GRADUL FUNCTIEI "
30  INPUT G
40  DIM C(G+1)
50  PRINT " DATI COEFICIENTII"
60  MAT INPUT C
70  DIM L(2)
80  PRINT " DATI DOMENIUL DE DEFINITIE "
90  MAT INPUT L
100 PRINT " DATI NUMARUL DE PASI "
110 INPUT H
120 IF L(1) < L(2) THEN 150
130 PRINT " LIMITE ERONATE "
140 GOTO 80
150 H = (L(2)-L(1))/N
160 X = L(1)
170 GOSUB 500
180 P = F
190 R = F
200 FOR K = 1 TO N
210 X = X+H
220 GOSUB 500
230 IF F >= P THEN 260
240 P = F
250 GOTO 280
260 IF F <= R THEN 280
270 R = F
280 NEXT K
290 IF R > P THEN 320
300 PRINT " FUNCTIE CU VALOARE CONSTANTA ";
305 PRINT " IN INTERVALUL DAT = ";R
310 STOP
320 INITP
330 WINDOW L(1),L(2),P,R
340 MOVE L(1),O
350 DRAW L(2),O
360 MOVE O,P
370 DRAW O,R
380 X = L(1)
390 GOSUB 500
400 MOVE X,F
410 FOR K = 1 TO N
420 X = X+H
430 GOSUB 500
440 DRAW X,F
450 NEXT K
460 STOP
500 REM " CALCULUL VALORII FUNCTIEI "
510 F = C(1)
520 FOR I = 2 TO G
530 F = F*X+C(I)
540 NEXT I
550 RETURN
560 END
```

13.7. Suma grafică a mai multor vectori

Se solicită prin dialog numărul total N al vectorilor de însumat, mărimea fiecăruia, și unghiul lor față de orizontală.

Masivul L (N) conține lungimile, iar A (N) unghiurile în grade.

Pentru fiecare vector se calculează coordonatele vîrfului vectorului considerînd vectorul din originea axelor:

$$X(I) = L(I)*\cos(A(I)*PI/180)$$

$$Y(I) = L(I)*\sin(A(I)*PI/180)$$

Coordonata vîrfului vectorului sumă se obține din formulele:

$$A = \sum_{i=1}^{N} X(I)$$

$$B = \sum_{i=1}^{N} Y(I)$$

iar lungimea vectorului sumă și unghiul cu orizontala:

$$L = \sqrt{A^2 + B^2}$$

$$C = (\mathrm{arctg}\,(B/A))*180/PI$$

```
10  PRINT " ADUNAREA VECTORILOR "
20  PRINT " NR. VECTORI DE ADUNAT "
30  INPUT N
40  DIM L(N),A(N),X(N),Y(N)
50  PRINT " DATI MARIMEA SI UNGHIUL "
55  PRINT " CU AXA X A VECTORILOR "
60  FOR I = 1 TO N
70  INPUT L(I),A(I)
80  NEXT I
90  FOR I = 1 TO N
100 X(I) = L(I)*COS(A(I)*PI/180)
110 Y(I) = L(I)*SIN(A(I)*PI/180)
120 NEXT I
130 A = X(1)
140 B = Y(1)
150 FOR I = 2 TO N
160 A = A+X(I)
170 B = B+Y(I)
180 NEXT I
185 INITP
190 L = SQR(A^2+B^2)
200 C = ATN(B/A)
210 PRINT AT(1,1);" LUNGIMEA ";L
220 PRINT AT(2,1);" UNGHIUL ";C*180/PI
230 WINDOW -1,A+1,-1,B+1
240 MOVE -1,0
250 DRAW A+1,0
260 MOVE 0,-1
270 DRAW 0,B+1
280 MOVE 0,0
290 FOR I = 1 TO N
300 RDRAW X(I),Y(I)
310 NEXT I
320 MOVE 0,0
330 DRAW A,B
340 STOP
350 END
```

După inițializarea ecranului, spațiul utilizator se stabilește în funcție de coordonatele A, B, se trasează axele și prin bucla 290—310 utilizînd instrucțiunea RMOVE vectorii de adunat avînd coordonatele relative X (I), Y (I). Se trasează și vectorul sumă.

13.8. Mişcarea unui punct material într-un cîmp gravitaţional

Se consideră un punct material care pleacă cu viteza inițială V_0 dintr-un punct A și trebuie să atingă un punct B aflat la o distanță D față de A. Mișcarea punctului este dată de ecuația:

$$Y = X \operatorname{tg} \omega - X2 \frac{5}{V_0^2 \cos^2 \omega}$$

unde:

V_0 este viteza inițială

ω este unghiul de aruncare față de orizontală.

Se consideră de asemenea că între punctele A și B poate exista o diferență de nivel N (punctele nu se află obligatoriu ambele pe axa orizontală).

Programul solicită prin dialog distanța între cele doua puncte, diferența de nivel între cele două puncte, viteza inițială a mobilului și unghiului său de aruncare.

```
10  PRINT " BALLISTIC "
20  PRINT " DISTANTA DE TRAGERE "
30  INPUT D
40  PRINT " DIFERENTA DE NIVEL "
45  PRINT " POZITIV SAU NEGATIV "
50  INPUT N
60  N = N*44/D
70  INITP
80  X = 10
90  Y = 10
100 V = 54
110 Z = 10
120 IF N >= 0 THEN 150
130 Y = Y+ABS(N)
140 GOTO 160
150 Z = Z+N
160 WINDOW 0,64,0,64
170 PLOT X,Y
180 PLOT V,Z
190 MOVE X,Y
200 PRINT AT(29,2);" VITEZA INITIALA "
210 INPUT W
220 PRINT AT(30,2);" UNGHIUL IN GRADE "
230 INPUT U
240 U = PI*U/180
250 T = TAN(U)
260 C = 5/(W*COS(U))^2
270 I = 0
280 FOR K = 0 TO 44
290 A = (K*T-C*K^2)
300 DRAW X+K,Y+A
310 IF K < 44 THEN350
320 IF ABS(Z-A) >= 1 THEN 350
330 PRINT AT(30.2)." LOVIT"
340 I = 1
350 NEXT K
360 IF I = 1 THEN 10
370 GOTO 190
380 END
```

Se trasează graficul traiectoriei punctului, specificîndu-se la sfîrşit dacă s-a atins sau nu punctul B.

Dacă nu a fost atins punctul B se cere introducerea noilor valori de viteză iniţială a mobilului şi a unghiului de aruncare pînă se găseşte combinaţia adecvată.

13.9. Generarea şi modificarea unei figuri

Programul generează desenul unui scaun văzut în perspectivă (fig. 13.5), solicitînd prin dialog dimensiunile l_1, l_2, l_3 şi l_4. După generare figura poate fi modificată succesiv, schimbînd una sau mai multe dimensiuni. Se pot obţine astfel o serie de reprezentări ale aceluiaşi obiect, cu diferite proporţii între dimensiunile principale.

Programul poate fi extins cu uşurinţă pentru generarea de figuri mai complicate, generarea mai multor figuri la diferite distanţe între ele, permutarea lor etc. ajutînd la proiectarea corpului respectiv.

```
5   PRINT " PROIECTARE ASISTATA "
10  PRINT " DATI LUNGIME,LATIME SEZUT SCAUN "
15  INPUT L1,L2
20  PRINT " DATI INALTIME SPATAR "
25  INPUT L3
30  PRINT" DATI INALTIME PICIOR "
35  INPUT L4
40  X1 = SQR(L1^2/2)
50  Y1 = X1
60  X2 = SQR(4*L2^2/5)
70  Y2 = X2/2
80  X3 = SQR(L3^2/4)
90  Y3 = 3*X3
100 A = X2+X1+X3+20
110 B = L4+Y2+Y1+Y3+20
120 INITP
130 WINDOW 0,A,0,B
140 MOVE 10,10
150 RMOVE 0,Y2
160 RDRAW 0,L4
170 RDRAW X1,Y1
180 RDRAW X2,-Y2
190 RDRAW -X1,-Y1
200 RDRAW 0,-L4
210 RDRAW 0,L4
220 RDRAW -X2,Y2
230 RDRAW X1,Y1
240 RDRAW X3,Y3
250 RDRAW X2,-Y2
260 RDRAW -X3,-Y3
270 RDRAW 0,-L4
280 PRINT " CONTINUATI (DA=1) "
290 INPUT C
300 IF C = 1 THEN 20
310 STOP
320 END
```

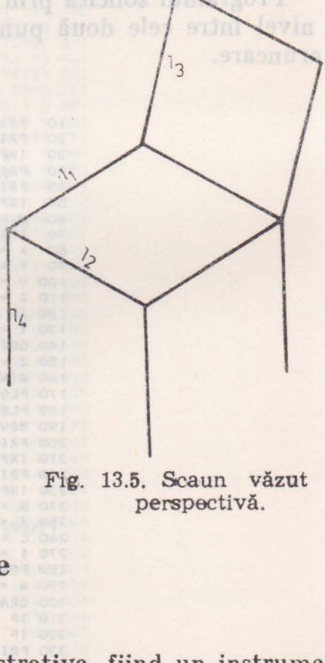


Fig. 13.5. Scaun văzut în perspectivă.

13.10. Generarea de figuri tridimensionale conform legilor perspectivei

Programul iese din sfera exemplelor demonstrative, fiind un instrument util pentru proiectarea asistată de calculator. Programul permite generarea de obiecte cu muchii drepte în spaţiul tridimensional marcat de axele OX,

OY, OZ. Obiectele se compun prin asocierea aşa-numitor segmente de dreaptă unitare. Lungimea unui segment unitar poate fi modificată pe parcurs. Indicarea direcţiei de trasare se face prin tastele asociate deplasărilor pe cele 3 axe de coordonate. Odată încheiată faza de generare a obiectului, acesta poate fi privit din orice punct din exteriorul său (ca şi cum privitorul s-ar roti în jurul obiectului). După dorinţă, obiectele generate pot fi completate sau modificate ulterior.

Imaginea în perspectivă a unui obiect a fost creată simulînd (aproximativ) fenomenul de creare a unei imagini pe retina ochiului. S-au luat în considerare următoarele ipoteze simplificatoare:

a) privirea este îndreptată totdeauna spre originea axelor de coordonate

b) cristalinul este o lentilă subţire convergentă, iar retina un plan ortogonal pe direcţia privirii, la distanţa de 3 cm faţă de centrul cristalinului.

S-a utilizat formula lentilelor convergente:

$$\frac{1}{0}+\frac{1}{i}=\frac{1}{f}$$

unde (fig. 13.6):

0 este distanţa între obiect şi lentilă (OB)
i este distanţa între obiect şi imagine (OB')
f este distanţa focală (OF)
A este punctul de proiectat
P este protecţia punctului A pe planul retinei
$\bar{i}, \bar{j}$ sînt versorii axelor de coordonate în planul retinei:

$$\bar{i}=\frac{\bar{n}\times \bar{e}_3}{\|\bar{n}\times \bar{e}_3\|}\ ;\ \bar{e}_3=(0,0,1)$$

$$\bar{j}=\frac{\bar{n}\times \bar{i}}{\|\bar{n}\times \bar{i}\|}.$$

Imaginea obţinută prin lentilă fiind răsturnată, la proiecţia pe ecran ea va fi din nou răsturnată pentru obţinerea imaginii corecte.

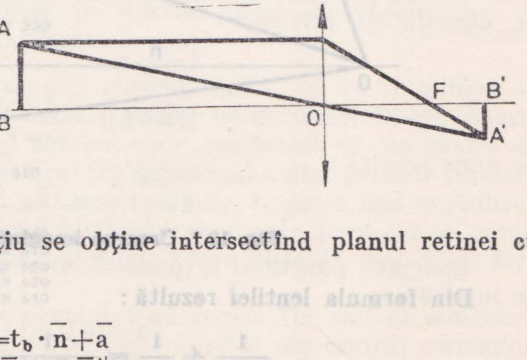


Fig. 13.6. Lentile convergente.

Poziţia punctului P în spaţiu se obţine intersectînd planul retinei cu dreapta AP:

$$\begin{cases} \bar{b}=t_b\cdot\bar{n}+\bar{a} \\ \bar{n}\cdot\bar{b}-\|\bar{n}\|^2=0 \end{cases}$$

iar poziția sa în plan:

$$\bar{p}=\bar{b}-\bar{n}=p^1\cdot\bar{i}+p^2\bar{j}\text{ ; rezultă}$$

$$p^1=-t.\bar{c}.\bar{i}\text{ ; }p^2=-t.\bar{c}.\bar{j}$$

Se obțin astfel formulele pentru calculul coordonatelor punctului pe planul retinei:

$$\bar{e3}=(0,0,1)$$

$$\bar{i}=\frac{\bar{n}\times\bar{e}_3}{\|\bar{n}\times\bar{e}_3\|},$$

$$\bar{\gamma}\quad\frac{\bar{n}\times\bar{i}}{\bar{n}\times\bar{i}\,\|},$$

$$t=\frac{3\,\|\bar{n}\|}{\|\bar{n}\|^2-\left(\frac{3}{\|\bar{n}\|}+1\right)\bar{n}.\bar{c}}.$$

$$p^1=-\bar{t}.c.\bar{i}$$

$$p^2=-\bar{t}.c.\bar{j}$$

Subrutina de la linia 2 000 din program aplică aceste formule la determinarea coordonatelor unui punct pe planul retinei.

Crearea imaginii unui punct oarecare notat cu C se va efectua conform figurii 13.7

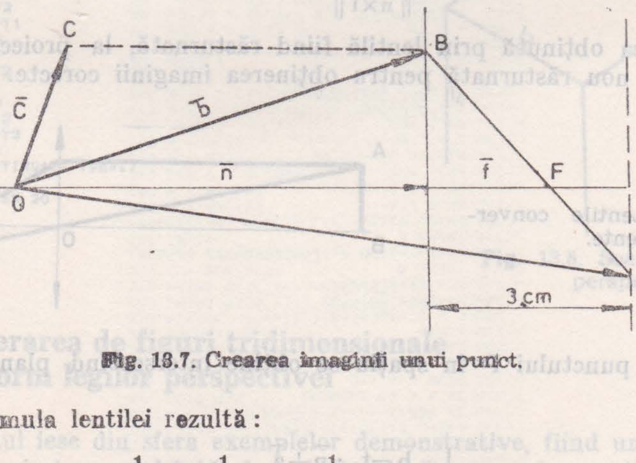


Fig. 13.7. Crearea imaginii unui punct.

Din formula lentilei rezultă:

$$\frac{1}{\|\bar{n}\|}+\frac{1}{3}=\frac{1}{\|\bar{n}-\bar{f}\|}$$

Punctul B se află la intersecţia planului lentilei cu raza de lumină paralelă cu direcţia privirii ce porneşte din C. Vom avea :

$$\begin{cases} \overline{b}=t_b\cdot\overline{n}+\overline{c} \\ \overline{n}\cdot\overline{b}-\|\,\overline{n}\,\|^2=0 \end{cases}$$

de unde rezultă :

$$\overline{b}=\frac{\|\,\overline{n}\,\|^2-\overline{n}\cdot\overline{c}}{\|\,\overline{n}\,\|^2}\cdot\overline{n}+\overline{c}.$$

```
5    INITP
10   DIM G$(2$4,12),R(254)
30   PRINT AT(2,1);" N = FIGURA NOUA "   *
40   PRINT AT(3,1);" V = FIGURA VECHE NEMODIFICATA "
50   PRINT AT(4,1);" M = FIGURA VECHE MODIFICATA "
60   INPUT F$
70   J = 1
75   E = 1
80   I = 0
82   U   0
84   V = 0
86   W = 0
90   GOSUB 5000
100  GOSUB 700
110  GOSUB 9000
120  GOSUB 500
130  GOSUB 3000
140  GOSUB 2000
150  MOVE C+50,D+50
160  GOSUB 9000
165  GOSUB 500
166  K$ = G$(I,JTOJ)
170  IF K$ = "A" THEN 160
180  IF K$ = "D" THEN 110
190  IF K$ = "S" THEN 250
200  IF K$ = "L" THEN 300
210  GOSUB 3u00
220  GOSUB 2000
230  DRAW C+50,D+50
240  GOTO 160
250  PRINT AT(1,1);" R = RELUATI
260  PRINT AT(2,1);" S = STOP
270  INPUT F$
280  IF F$ = "R" THEN 20
290  STOP
300  IF F$ = "N" THEN 330
310  G = R(E)
320  GOTO 370
330  PRINT AT(1,1);"
340  PRINT AT(1,1);
350  INPUT G
360  R(E) = G
370  E = E+1
380  GOTO 160
500  IF F$ <> M THEN 610
510  PRINT AT(1,1);"
520  PRINT AT(1,1);
530  A$ = INKEY$
540  IF A$ = " " THEN 530
550  IF A$ = "C" THEN 610
560  IF A$ = "I" THEN 600
570  IF A$ = "A" THEN 620
580  IF A$ = "D" THEN 640
590  GOTO 530
600  GOSUB 9040
610  RETURN
620  G$(I,JTOJ+2) = "AAA"
630  RETURN
```

```
640  F$ = "N"
650  RETURN
700  INITP
710  PRINT AT(27,26);"5=Z+'
720  PRINT AT(28,26);"6=Z-'
730  PRINT AT(29,26);"7=X+'
740  PRINT AT(30,26);"8=X-'
750  PRINT AT(31,26);"9=Y+'
760  PRINT AT(32,26);"0=Y-'
770  IF F$ <> "M" THEN820
780  PRINT AT(22,26);"C=CON'
790  PRINT AT(23,26);"I=INL'
800  PRINT AT(24,26);"A=ABN'
810  PRINT AT(25,26):"D=DEZ'
820  RETURN
2000 REM " CALCULE
2010 J3 = L*L+M*M
2020 IF J3 = 0 THEN 2080
2030 I1 = -M
2040 I2 = L
2050 J1 = -L*N
2060 J2 = -M*N
2070 GOTO 2120
2080 I1 = 0
2090 I2 = -1
2100 J1 = -1
2110 J2 = 0
2120 N1 = L*L+M*M+N*N
2130 IF N1 = 0 THEN 2290
2140 N1 = SQR(N1)
2150 N2 = I1^2+I2^2
2160 N2 = SQR(N2)
2170 N3 = J1^2+J2^2+J3^2
2180 N3 = SQR(N3)
2190 N4 = L*U+M*V+N*W
2200 N5 = N1^2-N4
2210 IF N5 = 0 THEN 2260
2220 T = 3*N1^2/N5
2230 C = -T*(U*I1+V*I2)/N2
2240 D = T*(U*J1+V*J2+W*J3)/N3
2250 RETURN
2260 C = 0
2270 D = 0
2280 RETURN
2290 PRINT " EROARE
2300 STOP
3000 FOR K = 0 TO 2
3010 K$ = G$(I,J+KTOJ+K)
3020 IF K$ = "5" THEN 3090
3030 IF K$ = "6" THEN 3110
3040 IF K$ = "7" THEN 3130
3050 IF K$ = "8" THEN 3150
3060 IF K$ = "9" THEN 3170
3070 IF K$ = "0" THEN 3190
3080 GOTO 3200
3090 W = W+G
3100 GOTO 3200
3110 W = W-G
3120 GOTO 3200
3130 U = U+G
3140 GOTO 3200
3150 U = U-G
3160 GOTO 3200
3170 V = V+G
3180 GOTO 3200
3190 V = V-G
3200 NEXT K
3210 RETURN
5000 IF F$ = "N" THEN 5030
5010 G = R(E)
5020 GOTO 5060
5030 PRINT AT(8,1);" MARIMEA ";
5040 INPUT G
5050 R(E) = G
5060 E = E+1
5070 PRINT AT(10,1);" COORDONATELE ":
5080 INPUT L,M,N
5090 RETURN
9000 I = I+1
9010 IF F$ = "N" THEN 9040
9020 REM " LA V SI M EXISTA IN MEMORIE "
9030 GOTO 9070
9040 PRINT AT (1,1);"              "
9050 PRINT AT(1,1);
9060 INPUT G$(I,JTOJ+2)
9070 IF I < 254 THEN 9130
9080 I = 0
9090 J = J+3
9100 IF J <= 9 THEN 9130
9110 PRINT " MEMMORY FULL "
9120 STOP
9130 RETURN
9140 END
```

Poziţia punctului F o aflăm din formula lentilei şi din faptul că F se află pe direcţia de privire:

$$\bar{f} = t_f \cdot \bar{n}$$

$$\| \bar{n} - \bar{f} \| = \frac{3 \| \bar{n} \|}{3 + \| \bar{n} \|}$$

de unde rezultă:

$$f = \left(1 + \frac{3}{3 + \| \bar{n} \|} \right) . \bar{n}.$$

Imaginea punctului C (punctul A) se obţine intersectînd cele două raze de lumină:

$$\begin{cases} \bar{a} = ta\,(\bar{f} - \bar{b}) + \bar{f} \\ \bar{a} = ta\,(\bar{n} - \bar{c}) + \bar{n} \end{cases}$$

Sistemul este compatibil determinat dacă vectorii $\{\bar{f}-\bar{b}, \bar{n}-\bar{c}\}$ sînt liniari independenți. Acest fapt este echivalent cu $\bar{c}\,(\bar{b}-\bar{n}) \neq 0$.

Deoarece $(\bar{b}-\bar{n}) \perp \bar{n}$ și $(\bar{b}-\bar{n}) \perp \bar{f}$, avem :

$$\bar{a} = t\,(\bar{n}-\bar{c}) + \bar{n} \text{ ; unde :}$$

$$t = \frac{3\,\|\bar{n}\|}{\|\bar{n}\|^2 - \left(\frac{3}{\|\bar{n}\|} + 1\right)\bar{n}\cdot\bar{c}}\,.$$

Folosind ecuațiile de mai sus, se creează imaginea obiectului care va trebui apoi proiectată pe planul retinei (fig. 13.8)

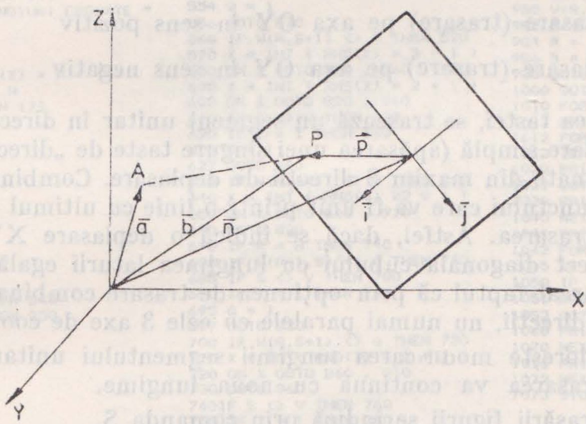


Fig. 13.8. Imaginea obiectului proiectată pe planul retinei.

Vom da în continuare instrucțiunile de exploatare ale programului. După lansarea sa în lucru, programul solicită operatorului una din opțiunile de bază :

a) N — generarea unei figuri noi

b) V — privirea figurii existente din diferite puncte de vedere

c) M — modificarea figurii existente

Opțiunea N Se cere lungimea segmentului unitar, ce va fi utilizat în trasările ulterioare. Se recomandă valori cuprinse în domeniul 5—2Ø. Aceste valori derivă din faptul că întreg ecranul este considerat ca un spațiu de 1ØØ×1ØØ unități. O valoare mai mică pentru segmentul unitar permite trasarea de figuri mai complexe. O valoare mai mare permite trasarea mai expeditivă a figurilor, riscîndu-se însă depășirea limitelor ecranului. În acest caz se modifică punctul de vedere al privitorului, plasîndu-l la o distanță mai mare față de obiect.

Se solicită apoi coordonatele punctului de vedere (în care se consideră că se află privitorul). Generarea figurii pornește implicit din centrul ecranului, considerat ca avînd coordonatele Ø, Ø, Ø. Prin comanda D (deplasare), se pot

modifica coordonatele de pornire a trasării. Aceeaşi comandă poate fi utilizată pentru părăsirea coordonatelor curente de trasare şi începerea trasării dintr-un alt punct. Coordonatele indicate în comanda D se raportează totdeauna relativ la coordonatele ultimului punct de trasare anterior.

Trasarea efectivă a cîte unui segment unitar se execută cu ajutorul următoarelor taste:

5— pentru deplasare (trasare) pe axa OZ în sens pozitiv

6— pentru deplasare (trasare) pe axa OZ în sens negativ

7— pentru deplasare (trasare) pe axa OX în sens pozitiv

8— pentru deplasare (trasare) pe axa OX în sens negativ

9— pentru deplasare (trasare) pe axa OY în sens pozitiv

Ø— pentru deplasare (trasare) pe axa OY în sens negativ

La apăsarea tastei, se trasează un segment unitar în direcţia specificată. În afară de trasare simplă (apăsarea unei singure taste de „direcţie") se permit şi trasări combinate din maxim 3 direcţii de deplasare. Combinaţia de direcţii indică poziţia punctului care va fi unit printr-o linie cu ultimul punct unde s-a oprit anterior trasarea. Astfel, dacă se indică o deplasare XYZ combinată, se va trasa direct diagonala cubului cu lungimea laturii egală cu segmentul unitar. De remarcat faptul că prin opţiunea de trasare combinată se pot face trasări în orice direcţii, nu numai paralele cu cele 3 axe de coordonate.

Dacă se doreşte modificarea lungimii segmentului unitar se utilizează comanda L. Trasarea va continua cu noua lungime.

Sfîrşitul trasării figurii se indică prin comanda S.

Opţiunea V Permite schimbarea poziţiei privitorului faţă de obiectul generat. Se introduc coordonatele noului punct de vedere, apoi programul trasează fără întrerupere imaginea vechiului obiect, văzut din noul punct dat.

Opţiunea M Permite modificarea facilă a imaginii unui obiect deja generat. Trasarea figurii se face pas cu pas, iar după fiecare pas se aşteaptă o comandă, care poate fi:

C — se trece la pasul următor;

I — se înlocuieşte comanda de trasare următoare cu ceea ce doreşte operatorul;

A — se abandonează comanda de trasare următoare (salt peste un pas);

D — dezvoltarea figurii — se consideră din acest moment trasarea ca şi la opţiunea N.

La terminarea trasării, operatorul poate relua ciclul celor trei opţiuni.

Domeniile de aplicaţie pot fi multiple, începînd cu exersarea vederii în spaţiu a diferitelor obiecte, şi terminînd cu activităţi de design în arhitectură, interioare de locuinţe, construcţii de maşini etc.

13.11. Trasare de labirint

Se trasează un labirint de lățime și lungime dată. Generarea se face utilizînd caracterele I, —, :, .. Astfel se poate genera labirintul fără utilizarea de instrucțiuni grafice. Labirintul are doar o singură intrare și ieșire și un singur drum de trecere. Fiind trasat pe baza generării de numere aleatoare se obțin forme diferite de labirint.

```
70 PRINT " LABIRINT "
80 DIM W(30,30),V(30,30)
90 PRINT " LATIMEA . LUNGIMEA "
100 INPUT H , V
102 IF H < 1 THEN 108
103 IF H < 30 THEN 140
104 IF V < 1 THEN 108
105 IF V < 30 THEN 140
106 PRINT " DIMENSIUNI ERONATE "
108 GOTO 90
140 Q = 0
150 Z = 0
160 X = INT( RND(X) * H + 1 )
165 FOR I = 1 TO H
170 IF I = X THEN 173
171 PRINT ".-";
172 GOTO 180
173 PRINT ". ";
180 NEXT I
190 PRINT "."
195 C = 1
196 W(X,1) = C
198 C = C+1
200 R = X
202 S = 1
205 GOTO 260
210 IF R <> H THEN 240
215 IF S <> V THEN 230
220 R = 1
222 S = 1
225 GOTO 250
230 R = 1
232 S = S+1
235 GOTO 250
240 R = R+1
250 IF W(R,S) = 0 THEN 210
260 IF R-1 = 0 THEN 530
265 IF W(R-1,S) <> 0 THEN 530
270 IF S-1 = 0 THEN 390
280 IF W(R,S-1) <> 0 THEN 390
290 IF R = H THEN 330
300 IF W(R+1,S) <> 0 THEN 330
310 X = INT( RND(X) * 3 + 1 )
320 ON X GOTO 790 , 820 , 860
330 IF S <> V THEN 340
334 IF Z = 1 THEN 370
338 Q = 1
339 GOTO 350
340 IF W(R,S+1) <> 0 THEN 370
350 X = INT( RND(X) * 3 + 1 )
360 ON X GOTO 790 , 820 , 910
370 X = INT( RND(X) * 2 + 1 )
380 ON X GOTO 790 , 820
390 IF R = H THEN 470
400 IF W(R+1,S) <> 0 THEN 470
405 IF S <> V THEN 420
410 IF Z = 1 THEN 450
415 Q = 1
416 GOTO 430
420 IF W(R,S+1). <> 0 THEN 450
430 X = INT(RND(X) * 3 + 1 )
440 ON X GOTO 790 , 860 , 910
450 X = INT( RND(X) * 2 + 1 )
460 ON X GOTO 790 , 860
470 IF S <> V THEN 490
480 IF Z = 1 THEN 520
485 Q = 1
486 GOTO 500
490 IF W(R,S+1) <> 0 THEN 520
500 X = INT( RND(X) * 2 + 1 )
510 ON X GOTO 790 , 910
520 GOTO 790
530 IF S-1 = 0 THEN 670
540 IF W(R,S-1) <> 0 THEN 670
545 IF R = H THEN 610
547 IF W(R+1,S) <> 0 THEN 610
550 IF S <> V THEN 560
552 IF Z = 1 THEN 590
554 Q = 1
556 GOTO 570
560 IF W(R,S+1) <> 0 THEN 590
570 X = INT ( RND(X) * 3 + 1 )
580 ON X GOTO 820 , 860 , 910
590 X = INT ( RND(X) * 2 + 1 )
600 ON X GOTO 820 , 860
610 IF S <> V THEN 630
620 IF Z = 1 THEN 660
625 Q = 1
627 GOTO 640
630 IF W(R,S+1) <> 0 THEN 660
640 X = INT ( RND(X) *2 + 1 )
650 ON X GOTO 820 , 910
660 GOTO 820
670 IF R = H THEN 740
680 IF W(R+1,S) <> 0 THEN 740
685 IF S <> V THEN 700
690 IF Z = 1 THEN 730
695 Q = 1
697 GOTO 830
700 IF W(R,S+1) <> 0 THEN 730
710 X = INT( RND(X) * 2 + 1 )
720 ON X GOTO 860 , 910
730 GOTO 860
740IF S <> V THEN 760
750 IF Z = 1 THEN 780
755 Q = 1
757 GOTO 770
760 IF W(R,S+1) <> 0 THEN 780
770 GOTO 910
780 GOTO 1000
790 W(R-1,S) = C
800 C = C+1
802 V(R-1,S) = 2
804 R = R-1
810 IF C = H*V + 1 THEN 1010
815 Q = 0
816 GOTO 260
820 W(R,S-1) = C
830 C = C+1
840 V(R,S-1) = 1
842 S = S-1
844IF C = H*V + 1 THEN 1010
850 Q = 0
854 GOTO 260
860 W(R+1,S) = C
870 C = C+1
872 IF V(R,S) = 0 THEN 880
875 V(R,S) = 3
877 GOTO 890
880 V(R,S) = 2
890 R = R+1
900 IF C = H*V + 1 THEN 1010
905 GOTO 530
910 IF Q = 1 THEN 960
920 W(R,S+1) = C
922 C = C+1
924 IF V(R,S) = 0 THEN 940
930 V(R,S) = 3
935 GOTO 950
940 V(R,S) = 1
950 S = S+1
952 IF C = H*V + 1 THEN 1010
955 GOTO 260
960 Z = 1
970 IF V(R,S) = 0 THEN 980
972 V(R,S) = 3
974 Q = 0
976 GOTO 1000
980 V(R,S) = 1
982 Q = 0
984 R = 1
986 S = 1
988 GOTO 250
1000 GOTO 210
1010 FOR J = 1 TO V
1011 PRINT "I ";
1012 FOR I = 1 TO H
1013 IF V(I,J) < 2 THEN 1030
1020 PRINT "  ";
1021 GOTO 1040
1030 PRINT "I ";
1040 NEXT I
1041 PRINT
1043 FOR I = 1 TO H
1045 IF V(I,J) = 0 THEN 1060
1050 IF V(I,J) = 2 THEN 1060
1051 PRINT ": ";
1052 GOTO 1070
1060 PRINT ":-";
1070 NEXT I
1071 PRINT "."
1072 NEXT J
1073 STOP
```

13.12. Mastermind

Jocul este o variantă adaptată a celebrului MASTERMIND, care solicită determinarea unei combinaţii de 4 culori într-o ordine anumită, din 6 culori posibile. Adaptarea s-a făcut pentru determinarea unei combinaţii de 4 cifre din 6 posibile. Setul de cifre posibile este : ,Ø 1, 2, 3, 4, 5. Se introduce o combinaţie de 4 cifre ; programul compară această combinaţie cu propria sa combinaţie stabilită prin randomizare la începutul jocului. Se afişează un "*" pentru cifră corectă în poziţie corectă, şi un „+" pentru cifră corectă în poziţie incorectă. Se admite ca aceiaşi cifră să apară de mai multe ori în cadrul combinaţiei de 4 cifre. Scopul jocului este de a determina combinaţia corectă din cît mai puţine încercări. Se afişează la sfîrşit numărul total de încercări.

```
5   INITP
10  PRINT " M A S T E R M I N D "
15  PRINT " GASITI PATRU CIFRE DIN "
20  PRINT " SASE POSIBILE 0,1,2,3,4,5 "
25  PRINT " RASPUNSUL LA O INCERCARE: "
30  PRINT " * = NR. BUN LA LOC BUN "
35  PRINT " + = NR. BUN LA ALT LOC "
40  L= O
45  DIM A(4),B(4)
50  FOR Z = 1 TO 4
55  Y = INT(RND(X)*6)
60  B(Z) = Y
65  NEXT Z
70  L = O
75  PRINT " DATI O INCERCARE "
80  L = L+1
85  MAT INPUT A
90  K = O
95  J = O
100 FOR Z = 1 TO 4
105 IF A(Z) <= 5 THEN 115
110 J = 1
115 NEXT Z
120 IF J = O THEN 135
125 PRINT " COMBINATIE ERONATA "
130 GOTO 85
135 FOR Z = 1 TO 4
140 B(Z) = ABS(B(Z))
145 IF A(Z) <> B(Z) THEN 165
150 K = K+1
155 A(Z) = 7
160 B(Z) = -B(Z)
165 NEXT Z
170 FOR Z = 1 TO 4
175 G = O
180 FOR H = 1 TO 4
185 IF A(H) = 7 THEN 215
190 IF A(H) <> B(Z) THEN 215
195 IF G <> O THEN 215
200 J = J+1
205 A(H) = 7
210 G = 1
215 NEXT H
220 NEXT Z
230 IF K = O THEN 250
235 FOR Z = 1 TO K
240 PRINT "*";" ";
245 NEXT Z
250 IF J = O THEN 270
255 FOR Z = 1 TO J
260 PRINT "+";" ";
265 NEXT Z
270 PRINT
275 IF K < 4 THEN 80
280 PRINT " ATI GASIT DIN ";L;
285 PRINT " INCERCARI "
290 PRINT " DORITI ALT JOC (DA/NU) "
300 INPUT D$
305 IF D$ = "DA" THEN 50
310 STOP
320 END
```

13.13. Vînătoarea de vulpi

Programul trasează un careiaj de N X N dimensiuni (N=10—30) în care plasează în mod aleator 3 vulpi. Scopul jocului este de a determina coordonatele vulpilor din cît mai puţine încercări. O încercare se specifică prin coordonatele punctului (X, Y), după care programul afişează un "/" pentru punct liber, sau un "X" pentru vulpe găsită. Se totalizează la sfîrşit numărul de încercări.

```
5   DIM A(6),C(3)
10  INITP
20  N = INT(RND(X)*30)
30  IF N >= 10 THEN 50
40  N = 10
50  FOR I = 1 TO 6
60  A(I) = INT(RND(X)*30)+10
70  IF A(I) < N THEN 100
80  A(I) = A(I)-5
90  GOTO 70
100 M = 0
110 IF I <= 2 THEN 170
120 FOR J = 1 TO I-1 STEP 2
130 IF A(I) <> A(J) THEN 150
140 M = 1
150 NEXT J
160 IF M <> 0 THEN 60
170 NEXT I
180 MAT C = ZER
190 M = 0
200 U = 0
205 V = 0
210 INITP
220 WINDOW -10,N+10,-10,N+10
230 PRINT AT(1,1);" VINATOARE DE VULPI "
240 PRINT AT(2,1);" GASITI TREI VULPI ";
245 PRINT "INTR-O ZONA DE ";N;"/";N
250 FOR I = 0 TO N
260 MOVE 0,I
270 DRAW N,I
280 MOVE I,0
290 DRAW I,N
300 NEXT I
310 PRINT AT(28,1);"                    "
320 PRINT AT(28,1);" COORDONATA ";
330 MAT INPUT B(2)
340 V = V+1
350 FOR J = 1 TO 5 STEP 2
360 IF A(J) <> B(1) THEN 530
370 IF A(J+1) <> B(2) THEN 530
380 I = INT(J/2)+1
390 IF C(I) = 0 THEN 420
400 PRINT AT(28,1);" VULPE DEJA GASITA "
410 GOTO 520
420 PRINT AT(28,1);" ATI GASIT O VULPE "
430 C(I) = 1
440 MOVE B(1),B(2)
450 DRAW B(1)+1,B(2)+1
460 MOVE B(1),B(2)+1
470 DRAW B(1)+1,B(2)
480 U = U+1
490 IF U <> 3 THEN 520
500 PRINT " ATI GASIT DIN ";V;
505 PRINT "INCERCARI "
510 M = 1
520 J = 5
530 NEXT J
540 IF M <> 0 THEN 20
550 MOVE B(1),B(2)
560 DRAW B(1)+1,B(2)+1
570 GOTO 310
580 END
```

13.14. Verificarea vitezei de reacţie

Programul afişează într-o poziţie aleatoare pe ecran o cifră (∅—9) şi aşteaptă un timp determinat introducerea aceleiaşi cifre de către operator. Se solicită iniţial viteza de joc (1—5), viteza 1 fiind cea mai mare. Se afişează permanent scorul pe ecran. Din start, jucătorul primeşte 5 puncte. Pe parcurs, modificarea scorului decurge astfel:

— pentru o cifră introdusă corect, se adaugă un punct ;

— pentru o cifră incorectă, se scade un punct ;

— pentru nici o cifră introdusă în intervalul de aşteptare, se scad 2 puncte.

Oprirea programului are loc la atingerea scorului nul.

```
5   INITP
10  PRINT "R E F L E X U L "
15  PRINT " SE AFISEAZA ALEATOR O CIFRA ";
20  PRINT "PE ECRAN "
25  PRINT " BATETI CIT MAI REPEDE TASTA ";
30  PRINT "RESPECTIVA "
35  PRINT " DIN START PRIMITI 5 PUNCTE "
40  PRINT " +1 PUNCT LA TASTA BUNA "
42  PRINT " -1 PUNCT LA TASTA ERONATA "
44  PRINT " -2 PUNCTE DACA NU TASTATI "
45  L = 5
50  PRINT " DATI VITEZA DE JOC (1,2,3,4,5) "
55  INPUT N
60  INITP
65  PRINT AT(1,1);" R E F L E X U L "
70  PRINT AT(2,1);" SCORUL: ";L
75  IF L < 1 THEN 220
80  A = INT(RND(X)*9)+48
85  B$ = INKEY$
90  X = INT(RND(X)*26+4)
95  Y = INT(RND(X)*29+1)
100 PRINT AT(X,Y);CHR$(A)
105 A$ = INKEY$
110 FOR I = 1 TO 100*N
115 IF A$ <> B$ THEN 125
120 B$ = INKEY$
125 NEXT I
130 IF B$ = A$ THEN 195
135 Q = 0
140 FOR I = 0 TO 9
145 IF B$ = STR$(I) THEN 155
150 GOTO 165
155 IF I <> A-48 THEN 165
160 Q = 1
165 NEXT I
170 IF Q = 1 THEN 185
175 L = L-1
180 GOTO 200
185 L = L+1
190 GOTO 200
195 L = L-2
200 FOR I = 1 TO N*100
205 REM " PAUSE "
210 NEXT I
215 GOTO 70
220 STOP
225 END
```

13.15. Perspico

Jocul constă în alinierea a trei 0-uri în linie sau diagonală, în cadrul unui careu de 3×3. Fiecare căsuţă din careu primeşte un număr (1—9), utilizat pentru indicarea locului în care se plasează 0-ul jucătorului. Ca răspuns, programul plasează un X într-o poziţie defavorabilă pentru jucător, căutînd la rîndul său să alinieze trei X-uri. La sfîrşit se afişează cîştigătorul.

```
5   PRINT " P E R S P I C O "
10  DIM B(9),P(9)
15  INITP
20  FOR A = 1 TO 9
25  B(A) = A
30  NEXT A
35  MAT P = ZER
40  E = 0
45  Q = 0
50  N = 0
55  X = 4
60  PRINT AT(X,3);
65  FOR A = 1 TO 9
70  IF A = X THEN 90
75  IF B(A) = 0 THEN 110
80  IF B(A) = 10 THEN 120
85  GOTO 130
90  X = X+3
95  PRINT
100 PRINT AT(X,3)+
105 GOTO 75
110 PRINT "0";" ";
115 GOTO 135
120 PRINT "X";" ";
125 GOTO 135
130 PRINT B(A);" ";
135 NEXT A
140 IF N = 1 THEN 210
145 IF E <> 8 THEN 160
150 PRINT AT(15,1);" AM CISTIGAT
155 GOTO 445
160 PRINT AT(15,1);" MUTAREA     "
165 INPUT Z
170 IF B(Z) <> Z THEN 165
175 N = 1
180 Q : Q+1
185 B(Z) = 0
190 P(Z) = 1
195 IF Q <= 0 THEN 55
200 PRINT AT(15,1);" REMIZA
205 GOTO 475
210 G = 0
215 C = 1
220 D = 7
225 F = 3
230 GOSUB 360
235 C = 2
240 D = 8
2455 GOSUB 360
250 C = 3
255 D = 9
260 GOSUB 360
265 C = 1
270 F = 4
275 GOSUB 360
280 D = 3
285 F = 1
290 GOSUB 360
295 C = 4
300 D = 6
305 GOSUB 360
310 C = 7
315 D = 9
320 GOSUB 360
325 C = 3
330 D = 7
335 F = 2
340 GOSUB 360
345 G = G+1
350 IF G = 5 THEN 50
355 GOTO 215
360 E = 0
365 U = 1
370 FOR A = C TO D STEP F
375 E = E+P(A)
380 NEXT A
385 IF E = 3 THEN 440
390 IF G = 0 THEN 435
395 IF E = 8 THEN 470
400 IF G = 1 THEN 435
405 IF E = 2 THEN 470
410 IF G = 2 THEN 435
415 IF E = 5 THEN 470
420 IF G = 3 THEN 435
425 IF E = 1 THEN 470
430 IF E = 4 THEN 470
435 RETURN
440 PRINT AT(15,1);" ATI CISTIGAT "
445 PRINT AT(17,1);" DORITI ALT JOC (DA=1) "
450 INPUT D
455 IF D = 1 THEN 15
460 STOP
470 FOR A = C TO D STEP F
475 IF B(A) <> A THEN 500
480 IF U = 0 THEN 500
485 B(A) = 10
490 P(A) = 4
495 U = 0
500 NEXT A
505 GOTO 50
510 END
```

13.16. Cursa de obstacole

Se conduce un mobil printr-o configurație de obstacole generată aleator pe ecran. Deplasarea mobilului are loc de la dreapta la stînga ; jucătorul are posibilitatea de a muta cu un rînd mai sus sau mai jos direcția de înaintare, apăsînd tasta S, respectiv J. Dacă nu se apasă nici o tastă, mobilul își continuă deplasarea în linie dreaptă. La lovirea unui obstacol, se oprește înaintarea și se afișează scorul. Valoarea scorului indică distanța parcursă pînă la lovirea obstacolului. După parcurgerea unui ecran complet, se reia mișcarea mobilului, modificînd locul de apariție al acestuia și combinația de obstacole.

```
10 PRINT " CONDUCETI VEHICOLUL PRINTRE ";
15 PRINT "OBSTACOLE "
20 PRINT " VEHICOLUL SE DIRIJEAZA CU ";
25 PRINT "TASTELE "
30 PRINT " S= S U S  ,  J= J O S "
40 DIM D(30,30)
50 S = 0
60 MAT D = ZER
70 INITP
80 FOR A = 1 TO 50
90 X = INT(RND(X)*29)+1
100 Y = INT(RND(X)*29)+1
110 PRINT AT(X,Y);"*"
120 D(X,Y) = 1
130 NEXT A
140 P = INT(RND(X)*29)+1
150 FOR A = 30 TO 1 STEP -1
160 PRINT AT(P,A);"<"
170 IF D(P,A) = 1 THEN 310
180 PRINT AT(P,A);"="
190 Z$ = INKEY$
200 IF Z$ = " " THEN 280
210 IF Z$ = "J" THEN 260
220 IF Z$ <> "S" THEN 280
230 IF P = 1 THEN 280
240 P = P-1
250 GOTO 280
260 IF P = 30 THEN 280
270 P = P+1
280 S = S+1
290 NEXT A
300 GOTO 60
310 FOR Z = 1 TO 20
320 PRINT AT(P,A);"*"
330 PRINT AT(P,A);" "
340 NEXT Z
350 PRINT AT(29,1);" SCORUL ";S
360 FOR I = 1 TO 100
370 REM " TIME "
380 NEXT I
390 GOTO 50
400 END
```

13.17. Tragerea la ţintă

Pe ecran se afişează o ţintă, considerată ca fiind în faţa privitorului. Se introduce viteza iniţială a proiectilului şi unghiul înălţătorului. Programul consideră ţinta ca fiind la o distanţă aleatoare faţă de trăgător. Din datele introduse, pe baza formulei :

$$S = V_0^2 \sin \omega / g$$

unde :

S este distanţa parcursă de proiectil;

V_0 este viteza iniţială ;

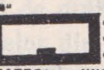
ω este unghiul înălţătorului ;

g este acceleraţia gravitaţională.

Se determină dacă s-a atins sau nu ţinta.

Dacă ţinta nu a fost atinsă, se afişează distanţa la care a căzut proiectilul faţă de ţintă („+" pentru lovitură prea lungă, „—" pentru lovitură prea scurtă). Se caută lovirea ţintei din cît mai puţine încercări, pe baza coroborării datelor afişate pentru loviturile în gol.

```
10  INITP
20  PRINT AT(1,10);" T I N T A "
30  PRINT AT(3,15);"▬"
40  PRINT AT(4,10);"
50  PRINT AT(5,10);"
60  PRINT AT(6,10);"
70  PRINT AT(8,1);" VITEZA    UNGHIUL  ";
75  PRINT "  DISTANTA "
80  PRINT AT(9,1);"---------------------";
85  PRINT "-----------"
90  K = 0
100 Q = 1
110 X = INT(RND(X)*15000+50)
120 K = K+1
122 IF K <=32 THEN 130
125 K = 32
130 PRINT AT(5,22);">";Q;"<"
140 PRINT AT(10+K,1);"?";
150 INPUT V
160 IF V <10 THEN 140
170 PRINT AT(10+K,10);">";
180 INPUT U
190 IF U < 3 THEN 170
200 IF U > 89 THEN 170
210 Z = INT(V^2*SIN(PI*U/90)/9.81)
220 E = Z-X
230 PRINT AT(10+K,20);E
240 IF Z = X THEN 270
250 Q = Q+1
260 GOTO 120
270 PRINT AT(13,1);" NIMERIT "
280 PRINT AT(15,1);" DORITI ALT JOC (DA=1) "
290 INPUT D
300 IF D = 1 THEN 10
310 STOP
320 END
```



13.18. Ecranul magic

Programul permite trasarea oricăror figuri compuse din drepte orizontale şi verticale, pornind dintr-un punct dat iniţial. Deplasarea punctului este coordonată de tastele :

1— dreapta	**3— jos**
2— stînga	**4— sus**

Punctul se deplasează continuu, oprirea trasării făcîndu-se prin apăsarea oricărei taste în afara celor de mai sus. Reluarea trasării se face prin apăsarea oricărei taste de comandă a direcţiei de deplasare.

Direcţia de deplasare poate fi completată, pentru a avea posibilitatea trasării sub un unghi oarecare faţă de orizontală. Dacă se doreşte o trasare la un unghi de 45° în direcţia dreapta-sus, se introduce linia 1Ø5 de recunoaştere a tastei "5" :

```
1Ø5 IF A$ ="5" THEN 5ØØ
```

La linia 5ØØ se introduce secvenţa de trasare :

```
5ØØ REM "DEPLASARE 45 DREAPTA SUS"
51Ø B$=INKEY$
52Ø IF Y=1ØØ THEN 57Ø
53Ø IF X=1ØØ THEN 57Ø
54Ø X=X+1
55Ø Y=Y+1
56Ø DRAW X, Y
```

```
570 A $ = INKEY $
580 IF A $ = B $ THEN 520
590 GOTO 60
```

Programul poate fi generalizat pentru trasări pe alte direcții, trasări de linii curbe etc.

```
5   INITP
10  PRINT AT(1,1);" ECRAN MAGIC "
15  PRINT AT(2,1);" 1= DREAPTA"
20  PRINT AT(3,1);" 2= STINGA "
25  PRINT AT(4,1);" 3= JOS "
30  PRINT AT(5,1);" 4= SUS "
35  PRINT AT(6,1);" COORD. DE INCEPUT ";
40  INPUT X,Y
45  MOVE X,Y
50  A$ = INKEY$
60  IF A$ = " " THEN 50
70  IF A$ = "1" THEN 120
80  IF A$ = "2" THEN 200
90  IF A$ = "3" THEN 280
100 IF A$ = "4" THEN 360
110 GOTO 50
120 REM " DEPL. DREAPTA "
130 B$ = INKEY$
140 IF X = 100 THEN 170
150 X = X+1
160 DRAW X,Y
170 A$ = INKEY$
180 IF A$ = B$ THEN 140
190 GOTO 50
200 REM " DEPL. STINGA "
210 B$ = INKEY$
220 IF X = 0 THEN 250
230 X = X-1
240 DRAW X,Y
250 A$ = INKEY$
260 IF A$ = B$ THEN 220
270 GOTO 50
280 REM " DEPL. JOS "
290 B$ = INKEY$
300 IF Y = 0 THEN 330
310 Y = Y-1
320 DRAW X,Y
330 A$ = INKEY$
340 IF A$ = B$ THEN 300
350 GOTO 50
360 REM " DEPL. SUS "
370 B$ = INKEY$
380 IF Y = 100 THEN 410
390 Y = Y+1
400 DRAW X,Y
410 A$ = INKEY$
420 IF A$ - B$ THEN 380
430 GOTO 50
990 END
```

13.19. Nim

Jocul Nim — originar din China antică — este cunoscut și sub numele de Fan-Tan.

Se generează într-un număr oarecare de grămezi un număr aleator de obiecte. Jucătorii (în cazul de față operatorul și calculatorul) elimină cu schimbul obiecte din grămezi, și anume cel puțin un obiect (cel mult toată grămada), dar la un moment dat numai dintr-o singură grămadă.

Cîștigă cine scoate ultimul sau ultimele obiecte din ultima grămadă rămasă.

O grămadă de obiecte se reprezintă printr-o linie de asteriscuri (un asterisc reprezintă un obiect). Numărul total de grămezi se stabilește de operator la începutul fiecărui joc. Numărul obiectelor din fiecare grămadă se generează aleator la începutul jocului.

```
10 DIM A(10),B(10,5),O(5),V(10)
15 PRINT " N - I - M "
20 PRINT " CU CITE GRAMEZI JUCATI? "
30 INPUT N
40 IF N <= 0 THEN 20
50 IF N >  9 THEN 20
55 MAT V = ZER
60 REM " GENERAREA GRAMEZILOR "
70 FOR I = 1 TO N
80 A(I) = INT( RND(X) * 30 )
65 IF A(I) = 0 THEN 80
90 NEXT I
95 GOSUB 2050
100 PRINT " DIN CARE GRAMADA SI CITE ";
105 PRINT " OBIECTE LUATI? "
110 INPUT I , K
120 IF I > N THEN 230
130 IF I < 0 THEN 230
140 IF K > A(I) THEN 230
150 IF K < 1 THEN 230
160 V(I) = K
170 GOSUB 650
180 REM " JOACA CALCULATORUL "
190 GOSUB 1300
200 GOSUB 650
210 IF C <> 0 THEN 290
220 GOTO 95
230 PRINT " EROARE "
240 GOTO 95
290 IF C = -1 THEN 320
300 PRINT " AM CISTIGAT "
310 STOP
320 PRINT " AI CISTIGAT "
330 STOP
650 FOR I = 1 TO N
660 A(I) = A(I) - V(I)
670 V(I) = 0
680 NEXT I
690 RETURN
1300 MAT B = ZER
1350 MAT O = ZER
1360 FOR U = 1 TO N
1390 M = A(U)
1400 FOR I = 1 TO 5
1410 B(U,I) = M - INT( M/2 ) * 2
1420 M = INT( M/2 )
1430 O(I) = B(U,I) + O(I)
1440 IF O(I) > 1 THEN 1460
1450 GOTO 1470
1460 O(I) = 0
1470 NEXT I
1475 NEXT U
1480 W = 0
1485 FOR U = 5 TO 1 STEP -1
1490 IF O(U) <> 0 THEN 1510
1500 GOTO 1530
1510 W = U
1520 U = 1
1530 NEXT U
1540IF W <> 0 THEN 1570
1550 GOSUB 1600
1560 RETURN
1570 GOSUB 1730
1580 RETURN
1600 W = 0
1610 FOR J = N TO 1 STEP -1
1620 IF A(J) <> 0 THEN 1640
1630 GOTO 1660
1640 W = J
1650 J = 1
1660 NEXT J
1670 IF W <> 0 THEN 1700
1680 C = -1
1690 RETURN
1700 C = 0
1705 J = W
1710 V(J) = A(J) - INT((A(J)-1)*RND(X))
1720 RETURN
1730 U = 0
1740 FOR I = 1 TO N
1745 IF B(I,W) <> 1 THEN 1760
1750 B(I,W) = 0
1752 U = I
1755 I = N
1760 NEXT I
1770 FOR J = W-1 TO 1 STEP -1
1775 IF O(J) = 0 THEN 1790
1780 B(U,J) = 1 - B(U,J)
1790 NEXT J
1795 I = U
1800 FOR J = 5 TO 1 STEP -1
1810 V(I) = 2 * V(I) + B(I,J)
1820 NEXT J
1830 V(I) = A(I) - V(I)
1840 S = 0
1850 FOR J = 1 TO N
1860 S = S + A(J) - V(J)
1870 NEXT J
1880 IF S = 0 THEN 1910
1890 C = 0
1900 RETURN
1910 C = 1
1920 RETURN
2050 FOR I = 1 TO N
2060 IF A(I) = 0 THEN 2100
2070 FOR J = 1 TO A(I)
2080 PRINT "*";
2090 NEXT J
2100 PRINT
2110 V(I) = 0
2120 NEXT I
2130 RETURN
```

13.20. Turnurile din Hanoi

Jocul este cunoscut şi sub numele Acele Faraonului şi constă în a transfera un set de discuri aranjate în ordine descrescătoare a mărimii lor de pe primul ac pe al treilea ac, utilizînd ca element ajutător acul din mijloc. Regula

```
100 REM " TURNURILE DIN HANOI "
110 DIM T(10,3)
120 E = 0
130MAT T = ZER
140 PRINT " TURNURILE DIN HANOI "
150 PRINT " DISCURILE SE MUTA DIN STINGA "
155 PRINT " IN DREAPTA SI NU SE PUNE DISC "
160 PRINT " MAI MARE PE DISC MAI MIC "
200 PRINT " CU CITE DISCURI JUCATI? (MAX 10) "
210 INPUT S
230 M = 0
240 IF S > 0 THEN 260
250 GOTO 270
260 IF S <=10 THEN 350
270 E = E+1
280 IF E > 2 THEN 310
290 PRINT " DATI NUMAR CORECT 1 - 10 "
300 GOTO 200
310 PRINT " NU JUCATI CORECT "
320 STOP
350 PRINT " IN PROGRAM DISCURILE SINT BOTEZATE "
360 PRINT " PRIN NUMERE ZECIMALE 1 LA 10 "
370 PRINT " DACA JUCATI CU MAI PUTIN DECIT 10 "
380 PRINT " VETI JUCA CU DISCURI DE NUMERE CELE "
385 PRINT " MAI MARI "
380 PRINT " MUTATI DE PE ACUL 1 PE ACUL 3 "
400 Y = 10
410 D = 10
420 FOR X = S TO 1 STEP -1
430 T(Y,1) = D
440 D = D-1
450 Y = Y-1
460 NEXT X
470 GOSUB 1240
480 PRINT AT(27,1);" CARE DISC IL MUTATI? ";
490 E = 0
500 INPUT D
510 IF D > 10-S THEN 530
520 GOTO 540
530 IF D <= 10 THEN 580
540 PRINT " DISC INEXISTENT "
550 E = E+1
560 IF E > 1 THEN 310
570 GOTO 480
580 REM " VERIFICA DACA DISCUL E SUB ALT DISC "
585 A = 10
590 B = 3
595 FOR R = 1 TO 10
600 FOR C = 1 TO 3
605 IF T(R,C) <> D THEN 630
610 A = R
615 B = C
620 R = 10
625 C = 3
630 NEXT C
635 NEXT R
640 IF T(A-1,B) <> 0 THEN 670
665 GOTO 700
670 PRINT " DISCUL E SUB ALT DISC "
690 GOTO 480
700 E = 0
710 PRINT " PE CARE AC IL MUTATI? ";
720 INPUT N
730 IF (N-1)*(N-2)*(N-3) = 0 THEN 800
740 E = E+1
750 IF E > 1 THEN 310
760 PRINT " DATI NUMAR CORECT 1 , 2 , 3 "
770 GOTO 710
800 A = 0
805 FOR R = 1 TO 10
```

```
810 IF T(R,N) = 0 THEN 825
815 A = R
820 R = 10
825 NEXT R
830 IF A = 0 THEN 870
835 REM " VERIFICA SA NU SE PUNA DISC MAI "
836 REM " MARE PE MAI MIC "
840 IF D < T(A,N) THEN 870
850 PRINT " DISC MAI MARE PE MAI MIC NU SE POATE "
860 GOTO 480
870 FOR V = 1 TO 10
875 FOR W = 1 TO 3
880 IF T(V,W) = D THEN 890
885 GOTO 910
890 A = V
895 B = W
900 V = 10
905 W = 3
910 NEXT W
915 NEXT V
920 FOR U = 1 TO 10
925 IF T(U,N) = 0 THEN 950
930 X = U-1
935 U = 10
940 GOTO 960
950 X = 10
960 NEXT U
970 T(X,N) = T(A,B)
980 T (A,B) = 0
990 REM " TIPARESTE STAREA ACTUALA "
1000 GOSUB 1240
1010 M = M+1
1015 A = 0
1020 FOR R = 1 TO 10
1030 FOR C = 1 TO 2
1040 IF T(R,C) = 0 THEN 1060
1050 A = 1
1060 NEXT C
1070 NEXT R
1080 IF A = 0 THEN 1120
1090 IF M <= 2^S THEN 480
1100 PRINT " ATI DEPASIT NUMARUL DE PASI MINIM "
1105 PRINT " NECESARI "
1110 STOP
1120 PRINT " ATI REZOLVAT IN ";M;" PASI "
1140 PRINT " MAI JUCATI? (DA/NU) "
1150 INPUT A$
1160 IF A$ = "DA" THEN 120
1170 STOP
1240 INITP
1250 A = 26
1260 FOR K = 10 TO 1 STEP -1
1270 Z = 5
1280 FOR J = 1 TO 3
1290 IF T(K,J) = 0 THEN 1340
1300 PRINT AT(A,Z-INT(T(K,J)/2)+1);
1310 FOR V = 1 TO T(K,J)
1320 PRINT "*";
1330 NEXT V
1340 Z = Z+10
1350 NEXT J
1360 A = A-3
1375 PRINT
1380 RETURN
```

de bază a transferului specifică obligativitatea mutării unui disc mai mic peste unul mai mare, niciodată invers.

În jocul realizat pe calculator se permit maxim 10 discuri. Discurile sînt reprezentate prin asteriscuri. Numărul de asteriscuri care formează discul reprezintă mărimea discului. Paşii necesari pentru a termina jocul sînt 2^n-1, unde n reprezintă numărul de discuri în joc. Dacă jucătorul nu termină în număr de paşi necesari, se opreşte jocul.

13.21. Jocul cu trei grămezi

Se generează trei grămezi cu diferite numere de obiecte în fiecare grămadă. Jucătorul mută dintr-o grămadă oarecare într-o altă grămadă un obiect în contul unui obiect din a treia grămadă care se elimină din joc. Inițial cel puțin

```
5 INITP
10 PRINT " JOCUL CU TREI GRAMEZI "
12 PRINT " --------------------- "
15 PRINT " SE DAU TREI GRAMEZI PRIN "
20 PRINT " NUMERELE DE OBIECTE DIN ELE "
25 PRINT " JOCUL CONSTA IN A MUTA UN OBIECT "
30 PRINT " DINTR-O GRAMADA IN ALTA  IN "
35 PRINT " CONTUL UNUI OBIECT DIN A  TREIA  "
40 PRINT " CARE SE ELIMINA DIN JOC "
45 PRINT " SCOPUL ESTE DE A AJUNGE LA DOUA "
50 PRINT " GRAMEZI GOALE SI IN A  TREIA "
55 PRINT " SA RAMINE UN SINGUR OBIECT "
60 PRINT " GRAMEZILE SE INDICA PRIN 1 , 2 . 3 "
85 DIM T(3)
90 T(1) = INT( RND(X) * 19 + 1 )
95 T(2) = INT (RND(X) * 19 + 1 )
100 T(3) = INT ( RND(X) * 19 + 1 )
105 N = INT( RND(X) * 2 + 1 )
110 M = INT( RND(X) * 2 + 1 )
115 IF N = M THEN 110
120 P = 1
125 IF P = N THEN 140
130 IF P = M THEN 140
135 GOTO 150
140 P = P + 1
145 GOTO 125
150 IF INT(T(N)/2) = T(N)/2 THEN 165
155 IF INT(T(M)/2) = T(M)/2 THEN 190
160 GOTO 205
165 IF INT(T(M)/2) = T(M)/2 THEN 175
170 GOTO 205
175 IF INT(T(P)/2) <> T(P)/2 THEN 205
180 T(P) = INT( RND(X) * 19 + 1 )
185 GOTO 175
190 IF INT(T(P)/2) = T(P)/2 THEN 205
195 T(P) = INT( RND(X) * 19 + 1 )
200 GOTO 190
205 PRINT " INCEPEM UN JOC (DA = 1) "
210 INPUT D
215 IF D = 1 THEN 225
220 GOTO 210
225 INITP
230 I = 1
235 PRINT AT(I,5);T(1);"  :  ";T(2);"  :  ";T(3)
240 I = I + 1
245 IF I < 30 THEN 255
250 GOTO 225
255 PRINT AT(31,2);"                              "
260 PRINT AT(30,2);" DIN CARE - IN CARE "
265 PRINT AT(31 2);
270 INPUT D,E
272 IF D = E THEN 255
275 IF D > 3 THEN 255
280 IF E > 3 THEN 255
282 Q = 0
285 FOR J = 1 TO 3
290 IF D = J THEN 310
295 IF E <> J THEN 310
300 T(J) = T(J) + 1
305 GOTO 315
310 T(J) = T(J) - 1
312 IF T(J) = -1 THEN 390
315 NEXT J
316 IF Q = 1 THEN 410
320 K = 0
325 L = 0
330 FOR J = 1 TO 3
335 IF T(J) <> 0 THEN 350
340 K = K + 1
345 GOTO 355
```

```
350 L = T(J)
355 NEXT J
360 IF K <> 2 THEN 235
365 IF L = 1 THEN 380
370 PRINT AT(30,1);" NU ATI AJUNS LA REZULTAT BUN "
375 GOTO 90
380 PRINT AT(30,1);" ATI AJUNS LA REZULTAT BUN "
385 GOTO 90
390 END
```

în două grămezi trebuie să existe obiecte. Scopul este ca în final să rămînă un singur obiect într-o grămadă, celelalte să fie goale.

Se poate demonstra că dacă în toate trei grămezile există inițial obiecte în număr par sau impar (deci sînt de aceeași paritate) atunci jocul nu are soluție. Dacă la două grămezi paritatea este aceeași și a treia are altă paritate, jocul are soluție și în final acele grămezi vor fi goale care au avut aceeași paritate, rămînînd un obiect în grămada cu paritate diferită.

13.22. Ruleta

Un joc clasic de noroc este ruleta. Simularea pe calculator scoate în evidență doar țelul ca jucătorul cu suma de bani inițială să rămînă cît mai mult în joc, nu latura de noroc.

În cazul de față se generează aleator numărul și culoarea cîștigătoare. Jucătorul poate miza prin numere între 1 și 50 cu 5 pînă la 5ØØ de unități de bani cu mai multe mize. Inițial jucătorul are la dispoziție 1 000 de unități de bani.

Mizele sînt după cum urmează.

— pe un număr se poate miza cu numărul respectiv (1 pînă la 36), pe Ø sau ØØ se mizează cu numerele 49 respectiv 50. Cîștigul este de 35 de ori miza.

— pe numere între 1 și 12 ; 13 și 24 ; 25 și 36 se mizează cu codurile 37 ; 38 respectiv 39, cîștigul fiind dublu.

— pe numere în prima coloană (1, 4, 7, 10...) în a doua coloană (2, 5, 8, 11 ....) și în a treia coloană (3, 6, 9, 12...) se poate miza cu codurile 40 ; 41 respectiv 42, cîștigul fiind tot dublu.

— pe numere între 1 și 18 ; 19 și 36 se mizează cu codurile 43 respectiv 44 cîștigul fiind doar miza

— pe număr par sau impar se mizează cu codurile 45 respectiv 46 ; pe roșu sau negru cu codurile 47 respectiv 48, cîștigul fiind simplu.

Pentru fiecare miză făcută de jucător trebuie să se indice două numere:

— prima reprezintă codul (număr între 1 și 50)

— a doua reprezintă valoarea mizei).

```
10 DIM B(100),C(100),T(100),X(38),A(50)
20 MAT X = ZER
30 P = 1000
40 D = 100000
50 PRINT " CITE MIZE FACETI? "
60 INPUT Y
70 IF Y < 1 THEN 50
80 IF Y <> INT(Y) THEN 50
85 IF Y > 50 THEN 50
90 MAT A = ZER
100 FOR C = 1 TO Y
110  PRINT " M I Z A ";C;
120 INPUT X,Z
130 B(C) = Z
140 T(C) = X
150 IF X < 1 THEN 110
IF X>38 THEN 110
170 IF X <> INT(X) THEN 110
180 IF Z < 5 THEN 110
190 IF Z <> INT(Z) THEN 110
210 IF Z > 500 THEN 110
220 IF A(X) = 0 THEN 250
230 PRINT " ACEST COD A MAI FOST "
240 GOTO 110
250 A(X) = 1
260 NEXT C
270 PRINT
275 PRINT "***** SE INVIRTE ROATA *****"
276 PRINT
280 S = 1 + INT(38*RND(X))
290 X(S) = X(S)+1
300 IF S < 37 THEN, 360
310 IF S = 37 THEN 340
320 PRINT " CISTIGA 00 "
330 GOTO 490
340 PRINT " CISTIGA 0 "
350 GOTO 490
360 RESTORE
370 K = 0
380 FOR I = 1 TO 18
390 READ R
400 IF R <> S THEN 420
410 K = 1
420 NEXT I
430 IF K = 1 THEN 470
440 A$ = "NEGRU"
450 PRINT " CISTIGA ";S;" , ";A$
460 GOTO 490
470 A$ = " ALB "
480 GOTO 450
490 PRINT
500 FOR C = 1 TO Y
510 IF T(C) < 37 THEN 1400
520 ON T(C)-36 GOTO 560,660,710,760,840
525 ON T(C)-41 GOTO 920,1000,1070,1110
530 ON T(C)-45 GOTO 1150,1190,1290
540 GOTO 1400
560 REM " 1 - 12 (COD 37) 2:1 "
570 IF S <= 12 THEN 620
580 PRINT " PIERDERE ";B(C);" BANI CU MIZA "
590 D = D + B(C)
600 P = P - B(C)
610 GOTO 1540
620 PRINT " CISTIG ";B(C)*2;" BANI CU MIZA "
630 D = D - B(C)*2
640 P = P + B(C)*2
650 GOTO 1540
660 REM " 13 -24 (COD 38) 2:1 "
670 IF S >12 THEN 690
680 GOTO 580
690 IF S < 25 THEN 620
700 GOTO 580
710 REM " 25 - 36 (COD 39) 2:1 "
720 IF S > 24 THEN 740
730 GOTO 580
740 IF S < 37 THEN 620
750 GOTO 580
760 REM " PRIMA COLOANA (COD 40) 2:1 "
770 K = 0
780 FOR I = 1 TO 34 STEP 3
790 IF S <> I THEN 810
800 K = 1
810 NEXT I
820 IF K = 1 THEN 620
830 GOTO 580
840 REM " COLOANA A DOUA (COD 41) 2:1"
850 K = 0
860 FOR I = 2 TO 35 STEP 3
870 IF S <> I THEN 890
880 K = 1
890 NEXT I
900 IF K = 1 THEN 620
910 GOTO 580
920 REM " COLOANA A TREIA (COD 42) 2:1 "
930 K = 0
940 FOR I = 3 TO 36 STEP 3
950 IF S <> I THEN 970
960 K = 1
970 NEXT I
980 IF K = 1 THEN 620
990 GOTO 580
1000 REM " 1 - 18 (COD 43) 1:1 "
1010 IF S < 19 THEN 1030
1020 GOTO 580
1030 PRINT " CISTIG ";B(C);" BANI CU MIZA ";C
1040 D = D - B(C)
1050 P = P + B(C)
1060 GOTO 1540
1070 REM " 19 - 36 (COD 44) 1:1 "
1080 IF S < 19 THEN 580
1090 IF S < 37 THEN 1030
1100 GOTO 580
1110 REM " NR. PAR (COD 44) 1:1 "
1120 IF S/2 <> INT(S/2) THEN 580
1130 IF S < 37 THEN 1030
1140 GOTO 580
1150 REM " NR. IMPAR (COD 45) 1:1 "
1160 IF S/2 = INT(S/2) THEN 580
1170 IF S < 37 THEN 1030
1180 GOTO 580
1190 REM " ROSU (COD 47) 1:1 "
1200 RESTORE
1210 K = 0
1220 FOR I = 1 TO 18
1230 READ R
1240 IF S <> R THEN 1260
1250 K = 1
1260 NEXT I
1270 IF K = 1 THEN 1030
1280 GOTO 580
1290 REM " NEGRU (COD 48) 1:1 "
1300 RESTORE
1310 K = 0
1320 FOR I = 1 TO 18
1330 READ R
1340 IF S <> R THEN 1360
1350 K = 1
1360 NEXT I
1370 IF K = 1 THEN 580
1380 IF S > 36 THEN 580
```

```
1390 GOTO 1030
1400 REM " O SI 00 (COD 49 SI 50) 1:1 "
1410 IF T(C) <49 THEN 1490
1420 IF T(C) = 49 THEN 1450
1430 IF T(C) = 50 THEN 1470
1440 GOTO 580
1450 IF S = 37 THEN 1510
1460 GOTO 580
1470 IF S = 38 THEN 1510
1480 GOTO 580
1490 IF T(C) = S THEN 1510
1500 GOTO 580
1510 PRINT " CISTIGUL ";B(C)*35;" BANI CU MIZA ";C
1520 D = D - B(C) * 35
1530 P = P + B(C) * 35
1540 NEXT C
1550 PRINT " BANCA ";D;" BANI "
1560 PRINT " AVETI ";P;" BANI "
1570 IF P > 0 THEN 1600
1580 PRINT " NU MAI AVETI BANI "
1590 GOTO 1630
1600 IF D > 0 THEN 1630
1610 PRINT " S-A GOLIT BANCA "
1620 D = 101000
1630 PRINT " CONTINUAM? (DA/NU) "
1650 INPUT Y$
1660 IF Y$ = "DA" THEN 50
1660 STOP
1670 DATA 1,3,5,7,9,12,14,16,18,19,21,23,25,27
1675 DATA 30,32,34,36
1680 END
```

13.23. Trasarea bioritmului

Programul solicită pentru persoana căreia i se trasează bioritmul următoarele date :

— data naşterii ;

— data de la care se doreşte începerea trasării.

Se verifică dacă anul, luna şi ziua de naştere sînt corecte, dacă nu, se reia întrebarea. La fel se verifică data trasării şi dacă data trasării este după data naşterii.

Se calculează numărul de zile trecute de la data naşterii pînă la data trasării. Diferenţa de ani (U-A) între trasare şi naştere ; diferenţa de zile (X—Z) între zilele de trasare şi naştere ; anii bisecţi pentru care se adună cîte o zi (B) ; lunile între naştere şi trasare Σ (Zi) se iau în considerare la calculul zilelor. Formula de calcul este

$$N=(U-A)*365\pm \Sigma_{k}^{l}(Zi)+(X-Z)+B$$

Se trasează pe o perioadă de 15 zile, începînd cu data solicitată, bioritmul persoanei în cauză. Intervalul de 15 zile a fost ales pentru a avea o imagine suficient de bună (neînghesuită) pe ecran.

După iniţializarea ecranului se trasează axa pe care se marchează cele 15 zile. Subprogramul 1ØØØ calculează valoarea curbei sinus în funcţie de variabila I cu care se apelează. Pentru variaţia intelectului periodicitatea este 33 zile, pentru psihic 23 zile iar pentru fizic 28 zile. Curba intelect se marchează cu "+", cea pentru psihic cu "*" iar cea pentru fizic cu ".". Dacă cele 15 zile trec dintr-o lună în luna următoare se apelează subprogramul 13ØØ. Pe ecran se trasează 3 sinusoide cu caracterele de mai sus.

Programul poate fi reluat pentru altă dată de lansare a trasării bioritmului.

```
5    PRINT " B I O R I T M "
10   DIM T(12)
15   T(1) = 31
20   T(2) = 28
25   T(3) = 31
30   T(4) = 30
35   T(5) = 31
40   T(6) = 30
45   T(7) = 31
50   T(8) = 31
55   T(9) = 30
60   T(10)= 31
65   T(11)= 30
70   T(12)= 31
75   PRINT " DATA NASTERII "
80   PRINT " ANUL "
85   INPUT A
90   PRINT " ZIUA "
95   INPUT Z
100  PRINT " LUNA "
105  INPUT L
110  IF A > 1999 THEN 75
115  IF A < 1900 THEN 75
120  IF Z < 1 THEN 75
125  IF Z > 31 THEN 75
130  IF L < 1 THEN 75
135  IF L > 12 THEN 75
140  PRINT " DATA LANSARII ANALIZEI "
145  PRINT " ANUL "
150  INPUT U
155  PRINT " ZIUA "
160  INPUT X
165  PRINT " LUNA "
170  INPUT Y
175  IF U < A THEN 140
180  IF U > 1999 THEN 140
185  IF X < 1 THEN 140
190  IF X > 31 THEN 140
195  IF Y < 1 THEN 140
200  IF Y > 12 THEN 140
205  B = 0
210  C = A
215  IF INT(C/4) <> C/4 THEN 225
220  B = B+1
225  C = C+1
230  IF C <= U THEN 215
235  IF INT(A/4) <> A/4 THEN 250
240  IF L <= 2 THEN 250
245  B = B-1
250  IF INT(U/4) <> U/4 THEN 265
255  IF Y > 2 THEN 265
260  B = B-1
265  N = (U-A)*365+(X-Z)+B
270  IF Y > L THEN 295
275  FOR I = Y-1 TO L+1
280  N = N-T(I)
285  NEXT I
290 GOTO 310
295 FOR I = L+1 TO Y-1
300 N = N+T(I)
305 NEXT I
310 INITP
320 PRINT AT(15,1);"------------------";
325 PRINT "--------------"
330 FOR J = 2 TO 30 STEP 2
340 PRINT AT(15,J);"I"
350 NEXT J
360 ON Y GOTO 370,380,390,400,410,420
362 H = Y-6
365 ON H GOTO 430,440,450,460,470,480
370 PRINT AT(2,2);U;".";X;".";"IANUARIE"
375 GOTO 490
380 PRINT AT(2,2);U;".";X;".";"FEBRUARIE"
385 GOTO 490
390 PRINT AT(2,2);U;".";X;".";"MARTIE"
395 GOTO 490
400 PRINT AT(2,2);U;".";X;".";"APRILIE"
405 GOTO 490
410 PRINT AT(2,2);U;".";X;".";"MAI"
415 GOTO 490
420 PRINT AT(2,2);U;".";X;".";"IUNIE"
425 GOTO 490
430 PRINT AT(2,2);U;".";X;".";"IULIE"
435 GOTO 490
440 PRINT AT(2,2);U;".";X;".";"AUGUST"
445 GOTO 490
450 PRINT AT(2,2);U;".";X;".";"SEPTEMBRIE"
455 GOTO 490
460 PRINT AT(2,2);U;".";X;".";"OCTOMBRIE"
465 GOTO 490
470 PRINT AT(2,2);U;".";X;".";"NOEMBRIE"
475 GOTO 490
480 PRINT AT(2,2);U;".";X;".";"DECEMBRIE"
490 FOR K = 1 TO 21 STEP 10
500 IF X < 10 THEN 530
510 PRINT AT(16,K);X
520 GOTO 540
530 PRINT AT(16,K+1);X
540 X = X+5
550 IF X <= 28 THEN 700
560 IF Y <> 2 THEN 610
570 M = 29
580 IF INT(U/4) = U/4 THEN 690
590 M = 28
600 GOTO 690
610 IF X <= 30 THEN 700
620 M = 30
630 IF Y = 4 THEN 690
640 IF Y = 6 THEN 690
650 IF Y = 9 THEN 690
660 IF Y = 11 THEN 690
670 IF X = 31 THEN 700
680 M = 31
690 X = X-M
700 NEXT K
710 FOR E = 2 TO 30 STEP 2
720 J = 33
730 GOSUB 1000
740 PRINT AT(H,E);"+"
750 J = 23
760 GOSUB 1000
770 PRINT AT(H,E);"*"
780 J = 28
790 GOSUB 1000
800 PRINT AT(H,E);"."
810 N = N+1
820 X = X+1
830 IF T(Y) < G THEN 850
840 GOTO 860
850 GOSUB 1300
860 NEXT E
870 PRINT AT(27,2);"INTELECT = + "
880 PRINT AT(28,2);"PSIHIC   = * "
890 PRINT AT(29,2);"FIZIC    = . "
900 STOP
1000 F = INT(N-INT(N/J)*J)
1005 F = (F/J)*PI*2
1010 F = SIN(F)
1015 H = 15-INT(F*10)
1020 RETURN
1300 IF Y <> 2 THEN 1340
1305 IF INT(U/4) <> U/4 THEN 1340
1310 IF X <> 29 THEN 1340
1310 X = X+1
1320 Y = Y+1
1325 IF Y <= 12 THEN 1340
1330 Y = 1
1335 U = U+1
1340 RETURN
```

13.24. Dicţionar de sinonime

Programul permite crearea unui dicţionar de sinonime în două sau trei limbi.

La creare (lansare program prin RUN) se stabileşte dacă se creează în 2 sau 3 limbi şi limbile în care se creează. La lansare cu GOTO 1Ø se utilizează dicţionarul creat. În acest caz se poate căuta după oricare limbă (sau după oricare rubrică) un cuvînt obţinînd celelalte cuvinte (rubrici).

La lansare cu GOTO 1Ø mai există posibilitatea de completare a dicţionarului (fişierului) creat cu cuvinte noi.

Dacă la căutare nu se găseşte cuvîntul cerut, se revine la secvenţa de căutare sau de completare a dicţionarului.

Programul poate fi utilizat şi ca bloc-notes, cele 3 rubrici corespunzătoare celor 3 limbi putînd conţine orice informaţie (de ex. numele unei persoane, adresa şi numărul de telefon asociate).

```
10  PRINT " DICTIONAR DE SINONIME "
20  IF N = 0 THEN 400
30  REM " DICTIONAR CREAT "
40  PRINT " DATI CUVINTE NOI (DA=1) "
50  INPUT G
60  IF G = 1 THEN 400
70  REM " CAUTARE IN DICTIONAR "
80  PRINT "·DUPA CARE LIMBA CAUTATI? "
90  INPUT B$(TO)
100 A = 0
110 IF B$ = X$ THEN 180
120 A = L
130 IF B$ = Y$ THEN 180
140 IF M = 2 THEN 80
150 A = 2*L
160 IF B$ = Z$ THEN 180
170 GOTO 80
180 PRINT " DATI CUVINTUL IN LIMBA ";B$
190 INPUT C$(TO)
200 I = 1
210 IF C$ = D$(I,A+1TOA+L) THEN 260
220 I = I+1
230 IF I <= N THEN 210
240 PRINT " NU EXISTA CUVINTUL ";C$
250 GOTO 80
260 PRINT D$(I,TOL);"=",
270 IF M = 2 THEN 290
280 PRINT D$(I,L+1TO2*L);"=",
290 PRINT D$(I,1+(M-1)*LTOM*L)
300 GOTO 80
400 REM " CREARE SI COMPLETARE DICTIONAR "
410 IF N <> 0 THEN 520
420 PRINT " IN 2 SAU 3 LIMBI CREATI? "
430 INPUT M
440 DIM D$(40,M*15)
450 DIM C$(15)
460 PRINT " DATI LIMBILE DICTIONARULUI "
470 INPUT X$
480 INPUT Y$
490 IF M = 2 THEN 510
500 INPUT Z$
510 L = 15
520 IF N > 40 THEN 620
530 PRINT " DATI CUVINTUL IN LIMBA ";X$
540 INPUT D$(N,TO15)
550 PRINT " DATI CUVINTUL IN LIMBA ";Y$
560 INPUT D$(N,16TO30)
570 IF M = 2 THEN 600
580 PRINT " DATI CUVINTUL IN LIMBA ";Z$
590 INPUT D$(N,31TO45)
600 N = N+1
610 GOTO 520
620 STOP
630 END
```

13.25. Ordonarea unui set de informații

Programul posedă un set de date (în cazul de față datele constau din numele, greutatea și înălțimea unui grup de persoane) pe care le poate ordona după un criteriu dat.

În exemplul de față, ordonarea se face pentru ordinea alfabetică, după greutate sau după înălțime. După ordonare, se afișează lista de date ordonată.

```
5   PRINT " ORDONAREA DUPA ALFABET "
7   PRINT " GREUTATE SAU INALTIME  "
10  DIM A$(20,10),S(20),M(20)
15  N = 0
20  N = N+1
25  READ A$(N,TO)
30  READ M(N)
35  READ S(N)
40  IF A$(N) <> "STOP" THEN 20
45  PRINT " DATI CRITERIUL DE ORDONARE "
50  PRINT " 1 = ALFABET "
55  PRINT " 2 = GREUTATE "
60  PRINT " 3 = INALTIME "
65  INPUT O
70  REM " ORDONAREA "
75  C = 0
80  FOR K = 1 TO N-1
85  ON O GOTO 90,100,110
90  IF A$(K) > A$(K+1) THEN 120
95  GOTO 170
100 IF S(K) > S(K+1) THEN 120
105 GOTO 170
110 IF M(K) > M(K+1) THEN 120
115 GOTO 170
120 S = S(K)
125 S(K) = S(K+1)
130 S(K+1) = S
135 B$ = A$(K,TO)
140 A$(K) = A$(K+1,TO)
145 A$(K+1) = B$(TO)
150 M = M(K)
155 M(K) = M(K+1)
160 M(K+1) = M
165 C = C+1
170 NEXT K
175 IF C > 0 THEN 75
180 ON O GOTO 185,195,205
185 PRINT " NUME     ";" INALTIME  ";" GREUTATE "
190 GOTO 210
195 PRINT " NUME     ";" GREUTATE "
200 GOTO 210
205 PRINT " NUME     ";" INALTIME "
210 FOR I = 1 TO N
215 ON O GOTO 220,230,240
220 PRINT A$(I);" ";M(I);" ";S(I)
225 GOTO 245
230 PRINT A$(I);" ";S(I)
235 GOTO 245
240 PRINT A$(I);" ";G(I)
245 NEXT I
250 STOP
260 DATA "POPESCU",200,80
265 DATA "CONSTANTIN",175,65.4
270 DATA "BARNA",180,70
275 DATA "MARESCU",185,71
280 DATA "MICUTU",205,91
285 DATA "STOP",0,0
300 END
```

13.26. Decodificarea numerelor romane

Programul servește la transformarea numerelor romane în numere zecimale. Se introduce numărul roman sub forma unui șir de litere. Decodificarea se face în baza regulilor definite prin DATA în liniile 44Ø—55Ø din program. Regulile specifică succesiunea corectă de cifre romane ; o succesiune eronată este semnalată printr-un mesaj. Numărul decodificat este apoi afișat pe ecran.

După introducerea numărului roman în şirul T$ şi a lungimii numărului (din cîte litere se compune) în L, se citeşte prima literă şi se verifică dacă e din setul de litere permise într-un număr roman (M, D, C, L, X, V, I). Prima literă poate fi oricare dintre aceste litere. Aceasta e specificată în prima linie de DATA unde de exemplu 100002 înseamnă că dacă litera e M valoarea este de 1000 iar după M pot să apară litere după regula (02) care se găseşte în linia următoare de DATA. Valoarea-1 în DATA înseamnă că numărul respectiv nu poate apare, deci se semnalează succesiunea eronată.

```
10    REM   DECODIFICAREA NUMERELOR ROMANE "
20    DIM A(10,7),C$(7,1)
30    MAT READ A(11,7)
40    C$(1) = "M"
50    C$(2) = "D"
60    C$(3) = "C"
70    C$(4) = "L"
80    C$(5) = "X"
90    C$(6) = "V"
100   C$(7) = "I
110   DIM T$(20)
120   R = 1
130   M = 0
140   C = 0
150   P = 0
160   PRINT " DATI NUMARUL ROMAN
170   INPUT T$(TO)
180   L = 0
190   FOR I   1 TO 20
200   IF T$(ITOI) = '    THEN 220
210   L = L+1
220   NEXT I
230   FOR I = 1 TO L
240   K = 0
250   FOR J = 1 TO 7
260   IF T$(ITOI) <> C$(J) THEN 280
270   K = J
280   NEXT J
290   IF K <> 0 THEN 320
300   PRINT " NUMAR ROMAN ERONAT
310   GOTO 120
320   X = A(R,K)
330   IF X < 0 THEN 300
340   C = (1-ABS(SGN(K-P)))*(C+1)
350   IF C > 2 THEN 300
360   P = K
370   M = M+INT(X/100)
380   R = X-100*INT(X/100)
390   IF R <> 0 THEN 410
400   I = L
410   NEXT I
420   PRINT M
430   GOTO 120
440   REM " DECODIFICAREA CONDITIILOR PI
450 REM "  M   , D     C     L   X    V   I
460 DATA 100002,50003,10008,5005,1009,507,110
470 DATA 100002,50003,10008,5005,1009,507,110
480 DATA   -1  , -1  ,10008,5005,1009,507,110
490 DATA   -1    -1  ,10004,5005,1009,507,110
500 DATA  -1    -1     -1  ,5006,1009,507,110
510 DATA  -1    -1     -1     -1 ,1006,507,110
520 DATA  -1    -1     -1     -1 , -1  -1 ,107
530 DATA  80005,30005,10004,5006,1009,507,110
540 DATA  -1    -1     8007,3007,1006,507,110
550 DATA  -1  , -1   , -1  . -1   800,300,107
560 END
```

13.27. Mira de control AMIC

Programul generează o imagine de control asemănătoare mirei TV, cuprinzînd trasări de drepte orizontale şi verticale, a unui cerc circumscris unui pătrat, şi a setului de caractere, inclusiv cele semigrafice. Mira obţinută poate fi utilizată la reglajul calităţii imaginii TV.

```
10 REM " IMAGINE DE CONTROL "
20 INIT P
30 PRINT AT(2,1);"#### IMAGINE DE CONTROL ####"
40 WINDOW 0,150,0,160
50 MOVE 152,75
60 FOR I = 0 TO 2*PI STEP PI/10
70 DRAW 75+67*COS(I),75+67*SIN(I)
80 NEXT I
90 MOVE 0,142
100 DRAW 160,142
110 MOVE 0,132
120 DRAW 160,132
130 PRINT AT(5,1);"##"
140 PRINT AT(5,23);"##"
150 PRINT AT(6,1);"##"
160 PRINT AT(6,23);"##"
170 PRINT AT(29,1);"##"
180 PRINT AT(29,23);"##"
190 PRINT AT(30,1);"##"
200 PRINT AT(39,23);"##"
210 MOVE 0,117
220 DRAW 160,117
230 MOVE 0,35
240 DRAW 160,35
250 MOVE 0,20
260 DRAW 160,20
270 MOVE 0,9
280 DRAW 160,9
290 PRINT AT(7,7);"###########"
300 PRINT AT(8,7);"###########"
310 PRINT AT(9,7);"###########"
320 PRINT AT(26,7);"###########"
330 PRINT AT(27,7);"###########"
340 PRINT AT(28,7);"###########"
350 MOVE 0,100
360 DRAW 160,100
370 MOVE 0,50
380 DRAW 160,50
390 MOVE 0,75
400 DRAW 25,75
410 MOVE 128,75
420 DRAW 160,75
430 MOVE 17,0
440 DRAW 17,140
450 MOVE 25,0
460 DRAW 25,140
470 MOVE 128,0
480 DRAW 128,140
490 MOVE 137,0
500 DRAW 137,140
510 PRINT AT(14,6);"!""#$%&'()*+,./"
520 PRINT AT(16,5);"0123456789:;<=>?"
530 PRINT AT(18,5);"ABCDEFGHIJKLMNOP"
540 PRINT AT(20,5);"RSTUVWZ [\]^"
550 MOVE 41,0
560 DRAW 41,50
570 MOVE 41,100
580 DRAW 41,140
590 MOVE 107,0
600 DRAW 107,50
610 MOVE 107,100
620 DRAW 107,140
630 REM "STOP"
640 GOTO630
650 END
```

13.28. Ceas electronic

Programul solicită prin dialog ora exactă, după care afişează imaginea unui ceas cu arătătoarele în mişcare. În dreapta-jos a ecranului se afişează şi ora sub forma HHMM. Reglajul avansului se face în linia 25Ø din program. Constanta T are valoarea de aprox. 11 ØØØ, o valoare mai mare ducînd la "întîrzierea" ceasului.

```
10  PRINT " C E A S U L "
20  PRINT " DATI ORA DE PORNIRE "
30  INPUT H
40  PRINT " MINUTUL "
50  INPUT M
60  INITP
70  PRINT AT(6,13);"*"
80  PRINT AT(16,21);"*"
90  PRINT AT(26,13);"*"
100 PRINT AT(16,5);"*"
110 MOVE 52,52
120 G = (M*PI/30)
130 DRAW 52+30*SIN(G),52+30*COS(G)
140 MOVE 52,52
150 G = (H+M/60)*PI/6
160 DRAW 52+30*SIN(G),52+30*COS(G)
170 PRINT AT(30,24);H;":";M
180 M = M+1
190 IF M = 60 THEN 210
200 GOTO 250
210 M = 0
220 H = H+1
230 IF H <> 24 THEN 250
240 H = 0
250 T = 11000
260 FOR I = 1 TO T
270 NEXT I
280 GOTO 60
290 END
```

13.29. Anagrame

Programul solicită introducerea unui şir de max. 20 de caractere, afişînd apoi combinaţii aleatoare ale şirului.

Cuvîntul se introduce în variabila şir A$, iar lungimea şirului în N. În masivul A (N) se aleg aleator cifre între 1 şi N care determină litera a A (I)-a din cuvînt. După completarea lui A (N) se afişează literele cuvîntului în ordinea din A (N). După afişare se reia ciclul. Programul se continuă pînă la oprire cu CTRL/C, după care se poate relansa cu alt cuvînt (şir).

```
5   REM " A N A G R A M A "
10  DIM A$(30)
20  PRINT " DATI UN CUVINT "
30  INPUT A$(TO)
40  N = 0
50  FOR I = 1 TO 20
60  IF A$(ITOI) = " " THEN 80
70  N = N+1
80  NEXT I
90  DIM A(N)
100 FOR I = 1 TO 7
130 A(Z) = INT(RND(X)*N)+1
140 U = 0
150 FOR J = 1 TO Z-1
160 IF A(J) <> A(Z) THEN 180
170 U = J
180 NEXT J
190 IF U <> 0 THEN 130
200 NEXT Z
210 B$ = " "
220 FOR B = 1 TO N
230 B$ = B$+A$(A(B))
240 NEXT B
250 PRINT B$
260 NEXT X
270 GOTO 100
280 END
```

13.30. Bugetul cheltuielilor zilnice într-o familie

Programul solicită introducerea cheltuielilor efectuate pe zile ; noile cheltuieli se introduc după ultima zi completată într-o rulare anterioară. La cerere, se afișează totalul cheltuielilor și histograma cheltuielilor pe zile.

Dacă se introduc cheltuielile pentru primele zile se lansează programul cu RUN. Pentru completare de cheltuieli se lansează cu GOTO 2Ø. Histograma cheltuielilor se dă la cerere.

```
10  DIM A(31)
15  I = 1
20  PRINT " CITE ZILE INTRODUCETI? "
25  INPUT N
30  FOR J = 1 TO N
40  INPUT A(I)
50  I = I+1
60  NEXT J
70  PRINT " VRETI HISTOGRAMA DA/NU "
80  INPUT D$
90  IF D$ <> "DA" THEN310
100 C = A(1)
110 T = A(1)
120 FOR J = 2 TO I
130 D = C-A(J)
140 IF D >= 0 THEN 160
150 C = A(J)
160 T = T+A(I)
170 NEXT J
180 INITP
190 PRINT AT(1,1);" HISTOGRAMA CHELTUIELILOR "
195 PRINT AT(2,1);" PE ";I;" ZILE "
200 PRINT AT(3,1);" TOTAL CHELTUIELI ";T
210 WINDOW -10,110,-10,110
220 MOVE 0,-10
230 DRAW 0,100
240 MOVE -10,0
250 DRAW 100,0
260 FOR J = 1 TO I
270 A(J) = 100*A(J)/C
280 MOVE J*3,0
290 DRAW J*3,A(J)
300 NEXT J
310 STOP
320 END
```

13.31. Microfișier

Programul permite crearea și exploatarea unei microbaze de date generată sub forma unui tabel de articole. Fiecare articol poate conține mai multe cîmpuri de lungimi diferite.

La creare se definește baza de date, prin următorii parametri :

- nume ;
- număr maxim de articole (max. 254) ;
- număr cîmpuri/articol;
- denumirile și lungimile fiecărui cîmp.

În funcție de parametrii dați, se dimensionează tabelul ce va conține baza de date. Se verifică dublele definiții pentru numele asociate cîmpurilor.

Exploatarea bazei de date se execută cu ajutorul următoarelor comenzi :

a) introducere articol — se caută o linie (articol) vidă din tabel și se introduc valorile asociate fiecărui cîmp din articol. Introducerea unui articol

se execută prin afişarea numelui fiecărui cîmp şi introducerea conţinutului cîmpului de către operator.

Dacă numai există loc pentru un nou articol (tabel plin) se emite mesajul FIŞIER PLIN, după care se poate utiliza comanda de ştergere articol şi introducerea celui solicitat.

b) ştergere articole după conţinut — se şterg articolele care conţin informaţii identice într-un cîmp dat. Se solicită denumirea cîmpului şi conţinutul său. Se caută fiecare articol care conţine în cîmpul respectiv informaţia dată şi se şterge articolul respectiv.

c) listare articole — se listează articolele din cadrul fişierului.

d) modificare cîmpuri după conţinut — se solicită numele cîmpului, vechiul conţinut şi noul conţinut. Se listează fiecare articol ce conţine cîmpul cu informaţia identică cu „conţinut vechi" şi se întreabă operatorul dacă doreşte sau nu modificarea articolului.

```
5   PRINT " M I C R O F I S I E R "
10  IF P <> 0 THEN 215
15  REM " SE CREEAZA FISIER NOU "
20  PRINT " DATI NUME FISIER "
25  INPUT P$
30  P = 1
35  PRINT " DATI NUMAR MAXIM ARTICOLE "
40  INPUT T
45  PRINT " DATI NUMAR CIMPURI/ARTICOL "
50  INPUT N
55  DIM C$(N,10),L(N),S(T),S$(3,10),X$(10)
60  MAT S = ZER
65  PRINT " DATI DENUMIRE SI LUNGIME CIMP "
70  D = 0
75  FOR I = 1 TO N
80  INPUT C$(I,10)
85  INPUT L(I)
90  D = D+L(I)
95  NEXT I
100 D = 1
105 FOR I = 1 TO N-1
115 X$ = C$(I)
120 FOR C = I+1 TO N
125 IF X$ <> C$(C) THEN 135
130 D = -1
135 NEXT C
140 NEXT I
145 IF D = 1 THEN 160
150 PRINT " CIMPURI CU ACELASI NUME "
155 STOP
160 L = 0
165 PRINT " INTRODUCETI COMANDA "
170 PRINT " 1 = INTRODUCERE ARTICOL "
175 PRINT " 2 = STERGERE ART. DUPA CONTINUT "
180 PRINT " 3 = LISTARE ARTICOLE "
185 PRINT " 4 = MODIF.CIMPURI DUPA CONTINUT "
190 PRINT " 5 = LISTARE PT. MODIFICARI "
195 PRINT " 6 = TERMINARE SESIUNE "
200 INPUT C
205 ON C GOTO 215,325,395,450,540,155
210 GOTO 165
215 PRINT " SE LUCREAZA CU FISIERUL ";P$
220 REM " INTRODUCERE ARTICOLE "
225 IF L < T THEN 240
230 PRINT " FISIER PLIN "
235 GOTO 165
240 D = 1
245 IF D > T THEN 230
250 IF S(D) = 0 THEN 265
255 D = D+1
260 GOTO245
265 A = 0
270 FOR I = 1 TO N
275 PRINT C$(I)
280 INPUT F$(D,A+1TOA+L(I))
285 A = A+L(I)
290 NEXT I
295 S(D) - 2
300 L = L+1
305 PRINT " MAI INTRODUCETI (DA=1)
310 INPUT C
315 IF C = 1 THEN 245
320 GOTO 165
325 PRINT " SE LUCREAZA CU FISIERUL ";P$
330 REM " STERGERE ARTICOLE DUPA CONTINUT"
335 PRINT " DATI NUME SI CONTINUT CIMP "
340 PRINT " DUPA CARE SE STERG ARTICOLELE"
345 INPUT S$(1,10)
350 INPUT S$(2,10)
355 GOSUB 1000
360 IF I < 0 THEN 165
365 D = 1
370 GOSUB 1100
375 IF D > T THEN 165
380 S(D) = 0
385 L = L-1
390 GOTO 370
395 PRINT " SE LUCREAZA CU FISIERUL ";P$
400 REM " LISTARE FISIER "
405 FOR I = 1 TO N
410 PRINT C$(I);",":
415 NEXT I
420 PRINT
425 FOR I = 1 TO T
430 IF S(I) = 0 THEN 440
435 GOSUB 1200
440 NEXT I
445 GOTO 165
450 PRINT " SE LUCREAZA CU FISIERUL ";P$
455 REM " MODIF.CIMPURI DUPA CONTINUT"
460 PRINT " DATI NUME CIMP SI CONTINUT
465 PRINT " VECHI SI NOU"
470 INPUT S$(1,10)
475 INPUT S$(2,10)
480 INPUT S$(3,10)
485 GOSUB 1000
490 IF I < 1 THEN 165
495 D = 1
500 GOSUB 1100
510 IF D > T THEN 165
512 GOSUB 1200
515 PRINT " SE MODIFICA (DA=1)
520 INPUT C
525 IF C = 1 THEN 535
530 F$(D,A+1TOA+L(I)) = S$(3,
535 GOTO 500
540 PRINT " SE LUCREAZA CU FISIERUL ";P$
545 REM " LISTARE FISIER PT.MODIFICARE
550 D = 1
555 GOSUB 1200
560 IF D > T THEN 165
565 GOSUB 1300
```

```
570 D = D+1
575 GOTO 165
1000 REM " CAUTARE NUME CIMP "
1005 I = 1
1010 A = 0
1015 IF I > N THEN 1040
1020 IF C$(I) = S$(1) THEN 1050
1025 A = A+L(I)
1030 I = I+1
1035 GOTO 1015
1040 PRINT " NU EXISTA CIMP ";S$(1)
1045 I = -1
1050 RETURN
1100 REM " CAUTARE CONTINUT "
1105 IF D > T THEN 1130
1110 IF S(D) <> 2 THEN 1120
1115 IF F$(D,A+1TOA+L(I)) = S$(2) THEN 1130
1120 D = D+1
1125 GOTO 1105
1130 RETURN
1200 REM " TIPARIRE ARTICOL "
1205 B = 0
1210 FOR J = 1 TO N
1215 PRINT F$(D,B+1TOB+L(J)),",",
1220 B = B+L(J)
1230 NEXT J
1235 PRINT
1240 RETURN
1300 REM " MODIFICAREA "
1305 PRINT " AL CITELEA CIMP SE MODIF
1306 PRINT " 0 = NU SE MODIFICA "
1310 INPUT C
1315 IF C = 0 THEN 1360
1320 PRINT " DATI CONTINUT NOU "
1325 INPUT S$(3,10)
1330 B = 0
1332 IF C = 1 THEN 1305
1335 FOR J = 1 TO C-1
1340 B = B+L(J)
1345 NEXT J
1355 GOTO 1305
1360 RETURN
1365 END
```

e) Listare articole şi modificare — se listează pe rînd fiecare articol. Se întreabă dacă se doreşte modificarea unuia sau mai multor cîmpuri din articol. Procedura continuă pînă la sfîrşitul fişierului.

13.32. Universul Conway

Se poate studia prin aceasta creşterea, modificarea, înmulţirea şi moartea celulelor dintr-o colonie. Viaţa celulei este influenţată de celulele înconjurătoare. La început pe o zonă de N*M locaţii se generează celule vii conform coordonatelor date de utilizator. O locaţie poate avea două situaţii : conţine sau nu o celulă vie. Din starea iniţială în baza legilor geneticii se ajunge la generaţii următoare. Starea locaţiilor în următoarea generaţie poate fi:

1 — celula vie supravieţuieşte ;

2 — celula vie moare ;

3 — s-a generat o nouă celulă vie.

Cele trei stări se determină după următoarele reguli :

1. Supravieţuieşte fiecare celulă dacă are două sau trei celule vii vecine.

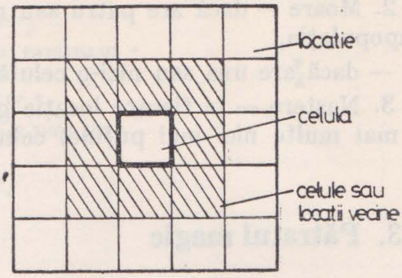


Fig. 13.9. Organizarea ecranului pentru universul lui Conway.

```
5 PRINT " UNIVERSUL CONWAY "
7 PRINT " ---------------- "
10 PRINT " DATI CIMPUL "
15 INPUT N,M
20 IF N > 30 THEN 10
25 IF M > 30 THEN 10
30 DIM U(N,M)
40 MAT U = ZER
50 PRINT " CITE CELULE VII INTRODUCETI? "
60 INPUT C
70 IF C <= N*M/2 THEN 100
90 GOTO 50
100 PRINT " DATI COORDONATELE CELULELOR VII "
110 FOR I = 1 TO C
120 INPUT X,Y
130 IF X > N THEN 120
140 IF Y > M THEN 120
150 IF X < 1 THEN 120
160 IF Y < 1 THEN 120
170 U(X,Y) = 1
180 NEXT I
185 INITP
190 PRINT AT(1,2);" STAREA INITIALA "
195 GOTO 220
200 INITP
210 PRINT AT(1,2);" GENERATIA URMATOARE "
220 FOR I = 1 TO N
230 PRINT AT(I+2,1);
240 FOR J = 1 TO M
250 IF U(I,J) = 0 THEN 280
260 PRINT "#";
270 GOTO 290
280 PRINT " ";
290 NEXT J
300 NEXT I
500 I = 1
510 FOR J = 1 TO N
520 C = 0
530 K = I + 1
540 IF K <= M THEN 560
550 K = M
560 L = I - 1
570 IF L >= 1 THEN 590
580 L = 1
590 FOR X = L TO K
600 O = J + 1
610 IF O <= N THEN 630
620 O = N
630 P = J - 1
640 IF P >= 1 THEN 660
650 P = 1
660 FOR Y = P TO O
670 IF INT( U(Y,X)/2) = U(Y<K)/2 THEN 690
680 C = C + 1
690 NEXT Y
700 NEXT X
710 D = 0
720 IF INT( U(J,I)/2) = U(J,I)/2 THEN 740
730 D = 1
740 C = C - D
750 IF C = 3 THEN 810
760 IF C <> 2 THEN 790
770 GOTO 830
790 D = D + 4
800 GOTO 820
810 D = D + 2
820 U(J,I) = D
830 NEXT J
840 I = I + 1
850 IF I <= M THEN 510
```

```
2000 FOR J = 1 TO M
2005 FOR I = 1 TO N
2010 IF U(I,J) > 3 THEN 2025
2015 IF U(I,J) > 1 THEN 2035
2020 GOTO 2045
2025 G = 0
2030 GOTO 2040
2035 G = 1
2040 U(I,J)   G
2045 NEXT I
2050 NEXT J
2055 GOTO 200
2060 END
```

2. Moare — dacă are patru sau mai multe celule vii vecine, motivul este suprapopulația,

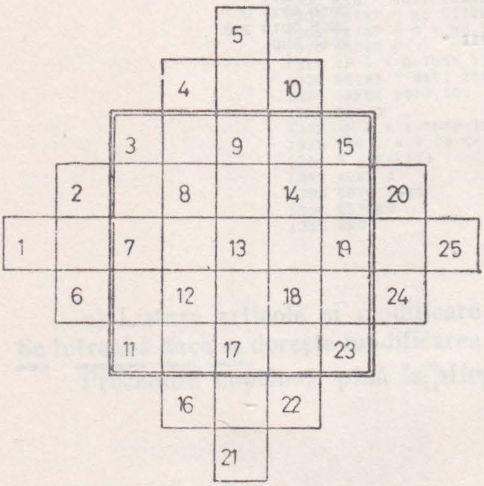
— dacă are una sau nici-o celulă vie vecină, motivul fiind izolarea.

3. Naștere — în fiecare locație goală se naște o celulă vie dacă are trei (nici mai multe nici mai puține) celule vii vecine.

13.33. Pătratul magic

Într-un pătrat de mărime dată se așază numere în așa fel încît pe orizontală pe verticală și în diagonală suma numerelor să fie aceeași. Există diferite metode de a genera pătratele magice de diferite mărimi.

Primul exemplu generează pătrate magice de grad impar utilizînd o generare bazîndu-se de următorul algoritm :



3	16	9	22	15
20	8	21	14	2
7	25	13	1	19
24	12	5	18	6
11	4	17	10	23

b

Fig. 13.10. Pătratul magic : (a) exemplu, (b) generale.

Pe pătratul de gradul dat se desenează piramide de pătrățele și în diagonalele astfel formate se trec cifrele în ordine crescătoare după care cifrele în afara pătratului se mută cu atîtea poziții cît este gradul pătratului.

Al doilea exemplu generează mai multe pătrate de gradul impar utilizînd următorul algoritm :

Din șirul de numere 1, 2, 3, 4, 5 ... pînă la gradul pătratului și Ø, n, 2*n, 3*n ... (unde n e gradul) pînă la (n—1)*n aleator se generează două pătrate de gradul dat sub forma :

— se alege din primul șir un șir aleator și se pune în primul rînd al pătratului ;

```
5 PRINT " PATRAT MAGIC "
7 PRINT " ------------- "
10 PRINT " DATI GRADUL PATRATULUI "
20 INPUT N
30 IF N > 0 THEN 70
40 PRINT " GRAD NEGATIV "
50 GOTO 20
70 IF INT( N/2 ) <> N/2 THEN 100
80 PRINT " GRAD PAR "
85 GOTO 20
90 IF N <= 9 THEN 100
95 PRINT " GRAD MARE "
97 GOTO 20
100 DIM M(N,N)
110 MAT M = ZER
120 J = (N+1)/2
130 I = J + 1
140 V = 1
150 IF M(I,J) = 0 THEN 210
160 IF V = N^2 + 1 THEN 460
170 I = I + 1
180 J = J - 1
190 IF I <= N THEN 150
200 GOTO 310
210 M(I,J) = V
220 IF V = N^2 THEN 400
230 V = V + 1
240 I = I + 1
250 J = J + 1
260 IF I > N THEN 300
270 IF J <= N THEN 290
280 J = 1
290 GOTO 150
300 IF J > N THEN 320
310 I = 1
315 GOTO 150
320 I = 2
330 J = N
340 GOTO 150
400 A = (30 - N*2)/2
402 A = A-3
410 FOR I = 1 TO N
415 B = A + I - 1
417 GOSUB 500
418 PRINT AT(B,B-1-(I-1)*3);"I"
420 FOR J = 1TO N
425 C = B+(J-1)*3-(I-1)*3
426 IF M(I,J) >= 10 THEN 430
427 C = C + 1
430 PRINT AT(B,C);M(I,J);"I"
440 NEXT J
445 A = A + 2
450 B = A + I - 1
460 GOSUB 600
465 PRINT
470 STOP
500 FOR K = 0 TO N
504 GOSUB 600
505 PRINT AT(B-1,B-1+3*K-(I-1)*3);"I"
510 NEXT K
515 RETURN
600 PRINT AT(B-2,B-(I-1)*3-1);
605 FOR L = 0 TO 3*N
610 PRINT "-";
615 NEXT L
620 RETURN
630 END
```

— restul liniilor se completează pornind cu numărul din pătratul de după mijlocul liniei și continuînd cu primele;

— din șirul al doilea de asemenea se alege o combinație aleatoare;

— restul liniilor se completează din acest șir dar acum pornind din mijlocul liniei.

Adunînd cele două pătrate se obține un pătrat magic:

2	1	4	3	5
3	5	2	1	4
1	4	3	5	2
5	2	1	4	3
4	3	5	2	1

Fig. 13.11. Primul pătrat.

5	0	20	10	15
20	10	15	5	0
15	5	0	20	10
0	20	10	15	5
10	15	5	0	20

Fig. 13.12. Al doilea pătrat.

7	1	24	13	20
23	15	17	6	4
16	9	3	25	12
5	22	11	19	8
14	18	10	2	21

Fig. 13.13. Pătratul rezultat.

```
10 PRINT " PATRAT MAGIC "
12 PRINT " ------------- "
20 DIM A(5),B(5),M(5),N(5),O(5),P(5)
30 DIM C(5,5),D(5,5),E(5,5)
40 MAT READ O
50 MAT READ P
60 FOR X = 1 TO 5
70 M(1) = INT( RND(X) * 5 ) + 1
80 FOR Z = 2 TO 5
90 M(Z) = INT( RND(X) * 5 ) + 1
100 U = 0
110 FOR J = 1 TO Z - 1
120 IF M(J) <> M(Z) THEN 140
130 U = J
140 NEXT J
150 IF U <> 0 THEN 90
160 NEXT Z
165 NEXT X
170 FOR X = 1 TO 5
180 N(1) = INT( RND(X) * 5 ) + 1
190 FOR Z = 2 TO 5
200 N(Z) = INT( RND(X) * 5 ) + 1
210 U = 0
220 FOR J = 1 TO Z - 1
230 IF N(J) <> N(Z) THEN 250
240 U = J
250 NEXT J
260 IF U <> 0 THEN 200
270NEXT Z
280 FOR J = 1 TO 5
290 A(J) = O(M(J))
300 B(J) = P(N(J))
310 NEXT J
315 NEXT X
320 FOR I = 1 TO 5
330 FOR J = 1 TO 5
340 C(I,J) = A(J)
350 D(I,J) = B(J)
360 NEXT J
370 M(1) = A(4)
380 M(2) = A(5)
390 M(3) = A(1)
400 M(4) = A(2)
410 M(5) = A(3)
420 N(1) = B(3)
430 N(2) = B(4)
440 N(3) = B(5)
450 N(4) = B(1)
460 N(5) = B(2)
470 MAT A = M
480 MAT B = N
490 NEXT I
500 MAT E = C + D
510 INITP
520 PRINT AT(2,2);" PATRAT MAGIC "
525 PRINT AT(3,2);" ------------ "
530 X = 0
540 FOR I = 1 TO 5
550 Y = 0
560 X = X + 4
570 FOR J = 1 TO 5
580 Y = Y + 4
585 Q = Y
590 IF E(I,J) > 9 THEN 610
600 Q = Q + 1
610 PRINT AT(X,Q);E(I,J)
620 NEXT J
630 NEXT I
640 GOTO 60
670 DATA 1,2,3,4,5
680 DATA 0,5,10,15,20
690 END
```

Capitolul 14.

Testarea resurselor hardware şi a interpretorului BASIC

Procedura de punere la punct a microcalculatorului urmează următoarea înlănţuire de faze distincte :

— pe placheta implantată cu componente electronice nealimentată se verifică — ohmetric — magistrala de adrese, date, comenzi ; scurtcircuite între alimentări (respectiv masă), cum şi restul pinilor de circuite ;

— pe placheta alimentată, dar fără circuitele ISI se verifică prezenţa tensiunilor, nivelul lor, cum şi modulele funcţionale, în următoarea ordine : prezenţa şi corectitudinea semnalelor delivrate de sincrogenerator, corectitudinea tastării RESET şi INI ;

— pe placheta alimentată cu circuitele LSI prezentate şi cu un EPROM de test programat cu NOP (de la adresa Ø) se verifică magistrala de adrese şi date ;

— se verifică corectitudinea semnalelor microprocesorului de comandă a memoriei, de selecţie etc. în conformitate cu diagrama de timp, înlocuind EPROM-ul de test cu un altul care asigură un program de scriere/citire în memoria RAM şi selecţia circuitelor EPROM de pe placheta ;

— cu EPROM-ul monitor implantat se verifică corectitudinea tastării, se reglează interfaţa cu receptorul TV şi casetofonul audio ;

— se trec testele hardware, se verifică comenzile de monitor şi setul de instrucţiuni BASIC.

14.1. Prezentare generală a setului de programe de test

Setul de programe realizat urmăreşte testarea logic-funcţională a modulelor harda microcalculatorului aMIC.

Programul TEST aMIC are următoarele module, corespunzătoare modulelor funcţionale pe care le testează :

R-RAM	Testarea zonei de memorie RAM
E-EPROM	Testarea zonei de memorie EPROM
D-DISPLAY	Testarea afişării pe ecran
K-TASTATURA	Testarea preluării de caractere de la tastatură.

Aceste programe pot fi executate în regim automat, ciclînd, fără intervenţia operatorului, sau în regim manual în care operatorul poate selecta testul dorit.

În afară de aceste teste există proceduri de testare a interpretorului BASIC şi a transferului de informaţii dinspre/spre casetofon.

Programul TEST aMIC este disponibil pe caseta magnetică, într-o variantă simplificată, sau este înscris pe EPROM. Completat cu cîteva subrutine modificate din monitor, formează conţinutul unei capsule 2716 de 2 Ko. Acest cip este amplasat în locul primului cip din EPROM-ul suport al programului BASIC. Programul utilizează generatorul de caractere, tabela de simboluri pentru tastatură şi cîteva subrutine din monitor.

Lansarea programului TEST AMIC se face prin intermediul monitorului, cu comanda

G∅ 8∅∅ (CR)

Această comandă cedează controlul monitorului de comenzi al programului de test. După ce se face iniţializarea stivei şi a variabilelor program (în zona de memorie RAM 5F∅∅H 5FFFH, corespunzătoare ultimei linii de caractere) se afişează mesajul:

TEST AMIC
MOD DE LUCRU : A SAU M.

Apăsarea unei alte taste decît A sau M va produce menţinerea mesajului de mai sus.

Următorul mesaj:

IN 2 CIP1 SAU 2?

cere numărul cipului de memorie EPROM plasat în poziţia 2. Această menţiune este necesară pentru textul EPROM, deoarece cipul care conţine TEST AMIC este plasat în poziţia 1. În funcţie de răspunsul la acest mesaj se va lua în considerare CHECKSUM-ul cipului 1 sau 2 la testarea poziţiei 2. din cele 8 EPROM-uri.

În funcţie de modul de lucru, tratarea se ramifică în continuare.

REGIMUL AUTOMAT: Se cere:

ORDINEA TESTELOR

care vor fi executate în regimul AUTOMAT.

În acest regim pot fi executate testele RAM, EPROM şi DISPLAY.

Ca răspuns la acest mesaj se introduce combinaţia dorită a testelor, permiţîndu-se şi repetarea unor teste cu condiţia ca numărul total să nu depăşească 1∅. Dacă succesiunea de caractere R, E şi respectiv D se termină cu C, atunci programul va cicla pe succesiunea de teste anterior introdusă. Comanda S intercalată în succesiunea de comenzi va avea ca efect terminarea testelor şi revenirea în monitorul microcalculatorului.

După introducerea celei de-a zecea comenzi, apare mesajul:

ZONE RAM TEST:

Acestea pot fi în funcţie de varianta microcalculatorului, următoarele: 6∅—∅∅; 6∅-C∅; 6∅—8∅; 4∅—8∅, A∅—∅∅ (memoria RAM alocată DISPLAY-

ului poate fi în zona 4Ø—6Ø sau 8Ø—AØ). În continuare vor fi executate testele înscrise în listă, în aceiaşi ordine. Dacă în lista testelor nu figurează comanda C atunci, după execuţie, se trece controlul monitorului de bază.

Regimul manual:

Pe ecran apare mesajul:

SIMBOL TEST ; aşteptîndu-se una dintre comenzile R, E, D sau K, care să lanseze programul test corespunzător. Comanda S trece controlul monitorului de bază iar orice tastă apărată reafişează mesajul de mai sus. De asemenea, după execuţia unui test se revine cu acelaşi mesaj (la dispoziţia operatorului).

14.2. Comanda E — testarea zonei de memorie EPROM

Testul se execută în mod identic în ambele regimuri de lucru. Acest test constă în calcularea sumei de control pentru fiecare cip de memorie EPROM-2716 şi compararea ei cu suma corespunzătoare anterior calculată şi înscrisă într-o listă de „semnături" în programul TEST aMIC.

Programul care urmează calculează sumele de control ale celor opt cipuri 2716 existente şi afişează valorile respective. Acest program se utilizează la fiecare modificare a informaţiei în memoria EPROM, valorile obţinute pentru sumele de control trebuind să fie introduse în tabelul de „semnături" din EPROM-ul de test.

```
CSMART:  MVI E,ØØ       ; Se încarcă în E nr. cipului
CS1:     MOV B,E        ; de început
         CALL CSC       ; Calcul suma de control a cipului respectiv
         MOV D,A        ; Se încarcă în D suma de control
         CALL CRLF      ; Se afişează nr. cipului
         MOV A,B
         CALL BINASC
         CALL AF2ØH     ; Se afişează suma de control
         MOV A,D
         CALL BINASC
         INR E          ; Increment nr. cipului
         CALL KEYIN     ; Aşteaptă comanda „continuă"
         CPI 43
         JZ CS1
         JMP MONIT.
```

Subrutina CSC calculează suma de control a cipului adresat

Intrare : B = nr. cipului adresat

Ieşire : A = suma de control

```
CSC:     PUSH B
         MOV A,B        ; calculul adresei de început
         RLC
         RLC
         RLC
```

```
        MOV H, A       ; (H, L) conţine adresa de început a cipului
        MVI L,ØØ
        ADI Ø8
        MOV B,A        ; B conţine adresa H de sfîrşit a cipului
        XRA A
CSC1 :  ADD M          ; calculul sumei de control
        MOV C,A
        INX H
        MOV A,H
        CMP B
        MOV A,C
        JNZ CSC1
        POP B
        RET
```

Deoarece cipul care conţine programele de test reamplasează în poziţia 1 (primul cip de BASIC), cipurile 1 şi 2 se testează pe rînd, amplasîndu-se în poziţia 2. Eventualele erori care apar se afişează sub forma : număr cip, sumă de control din tabel (martor), sumă de control calculată şi o tratare a cauzei apariţiei erorii respective.

Pentru testarea logicii de citire a memoriei EPROM se face, citirea şi calculul repetat al sumei de control pentru cipul care conţine monitorul. O primă măsură care se poate lua în cazul apariţiei de erori este mişcarea cipurilor în soclu, după care se va executa din nou testul EPROM în regim manual.

14.3. Comanda K — testarea preluării de caractere de la tastatură

Deoarece în cadrul acestui test este necesară intervenţia operatorului, testul se execută doar în regim manual. După afişarea mesajului de start test se aşteaptă acţionarea unei taste. Caracterul activat va fi pe o linie sau pe tot ecranul, funcţie de comanda precedentă.

CTRL-L – rămîne memorată şi este activată prin start test. Caracterul corespunzător tastei apăsate va fi afişat pe o linie, apăsarea tastei DEL are ca efect ştergerea ultimului rînd, iar tasta BS repetă ultimul caracter tastat.

CTRL-C – rămîne memorată pînă la acţionarea CTRL-L şi implică afişarea caracterului selectat pe întregul ecran. Tasta DEL şterge ecranul, iar tastele BS şi BLANK au acelaşi efect ca în regimul CTRL-L.

În ambele regimuri, acţionarea tastei CR transferă controlul monitorului TEST aMIC. Testul se bazează pe observaţiile operatorului. Se poate utiliza testul, prin menţinerea pe ecran a caracterului H sau a caracterelor semigrafice pentru reglarea calităţii imaginii ecranului monitorului (dimensiunile rastrului, liniarizate, focalizare, afişare video-invers stabilă).

14.4. Comanda D — testarea afişării pe ecran

Acest test verifică funcţionarea circuitelor de ceas şi formatoare a semnalului video-complex, cît şi a memoriei RAM din zona 4ØØØ-5FFFH rezervată afişării.

În primele 2 faze se afişează pe întregul ecran „table de şah" avînd dimensiunea pătratului elementar de una, respectiv patru, linii TV. Prima reţea se realizează prin înscrierea în RAM a octeţilor AAH şi, respectiv, 55H iar a doua prin afişarea caracterelor semigrafice avînd codurile 68H şi, respectiv, 6EH. După durate egale de timp imaginile sînt negativate, ceea ce se realizează prin complementarea informaţiei din RAM. Eventualele erori de înscriere în RAM la complementarea informaţiei sînt sesizate uşor pe DISPLAY de către operator. O caracteristică a „tablei de şah" este egalitatea între nivelul de alb şi de negru din imagine. Pentru testarea modulatorului s-au realizat succesiuni de imagini care conţin cantităţi diferite de alb şi de negru.

Astfel, în a treia fază se afişează pătrăţelele negre pe fond alb, utilizînd caracterele semigrafice (dimensiunea pătratului fiind de 8 linii TV), iar în a patra fază ecranul este şters (nivelul maxim de negru). Complementînd succesiv aceste imagini, nivelul de alb comută între 25% şi 75% în primul caz şi între Ø% şi 1ØØ% în al doilea caz. Pentru a obţine stabilitatea imaginilor se fac reglaje asupra modulatorului, prin modificarea punctelor de funcţionare a etajului modulator şi modificarea raportului între semnalele de sincronizare şi cel de imagine care concură la realizarea semnalului video-complex (SVC).

Pentru a testa adresarea intercalată a memoriei RAM-DISPLAY se afişează benzi verticale de contrast, de grosime echivalentă cu 8 linii TV. Benzile se afişează de la stînga la dreapta şi succesiv imaginea este complementată.

Pentru tastarea îndelungată a funcţionării microcalculatorului, cu afişarea pe ecran, s-au realizat programe în limbaj maşină sau BASIC de trasare a unor figuri geometrice (pătrate şi romburi înscrise, cercuri concentrice sau spirale). În continuare vor fi prezentate cîteva subprograme folosite pentru trasarea de figuri geometrice.

Considerîndu-se ecranul TV ca un pătrat cu latura de 256 linii TV şi notînd axa orizontală cu Ox iar cea verticală cu Oy, subprogramele testează, setează sau şterg orice punct TV de coordonate A (X, Y).

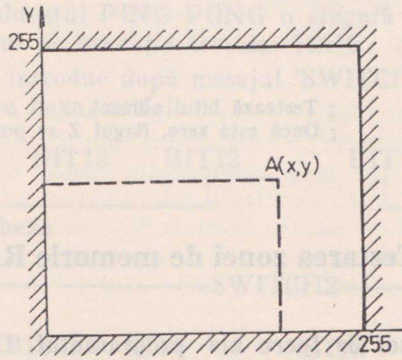


Fig. 14.1. Ecranul TV.

Subrutina ADR precizează poziţia bitului în memoria RAM-DISPLAY, corespunzător punctului A (X, Y).

Ca intrări se dau în registrele B şi C respectiv coordonatele X şi Y iar ca ieşiri se primeşte în registrul HL — adresa octetului şi în acumulator poziţia bitului căutat, codificată binar. Cunoscînd poziţia bitului în memoria RAM se pot opera asupra lui diferite acţiuni:

testare (TESTP), ştergere (RESETP) şi punere pe „1" (SETP)

```
ADR :    PUSH DE          ; Salvarea registrelor BC şi DE
         PUSH BC
         LD HL,3FE0H      ; Adresa început zonă RAM-DISPLAY
         LD DE,0020H        minus 20H
         INC C            ; DE= numărul de octeţi corespunzător unei linii TV
A1 :     ADD HL, DE
         DEC C
         IP NZ, A1        ; HL= adresa primului octet din rîndul TV căutat
         DEC HL.
A2 :     INC HL
         LD A,B
         SUB A,08H
         LD B,A
         JP NC A2         ; HL conţine adresa octetului căutat
         ADD A,08H ;      ; A= conţine nr. de ordine a bitului căutat în octet
         LD C,A
         INC C
         SCF              ; CY=''1''
         XOR A            ; A=00 ;
A3 :     RLA
         DEC C            ;
         JR NZ,A3         ; A= conţine un singur ''1'' în poziţia corespunzătoare
         POP BC             bitului căutat
         POP DE
         RET
SETP :   CALL ADR         ; Punerea pe ''1'' a bitului adresat de registrele B şi C.
         OR (HL)
         LD (HL),A
         RET
RESETP : CALL ADR         ; Ştergerea bitului adresat de registrele B şi C
         CPL
         AND (HL)
         LD (HL),A
         RET
TESP :   CALL ADR         ; Testează bitul adresat
         AND (HL)         ; Dacă este zero, flagul Z se poziţionează pe 1.
         RET
```

14.5. Comanda R — Testarea zonei de memorie RAM

14.5.1. Descrierea modurilor de lucru ale programului. După lansare calculatorul afişează mesajul:

TEST RAM

SWITCH: cerîndu-se 2 octeţi pentru precizarea modului de lucru. În regim automat cei doi octeţi care reprezintă SWITH SOFT vor fi înscrişi

automat cu valorile : SWC1=Ø1 ; SWC2=22 şi vor fi tipăriţi pe ecran. În regimul MANUAL vor fi preluaţi de la tastatură, biţii lor avînd următoarea semnificaţie :

BIT0 —Ø—
1 execută şi testul complementar (subtestul adresare)
BIT1 —Ø—
1 Oprire la eroare : continuă cu comanda Z.
BIT2 —Ø—
1 Ciclează pe locaţia unde a apărut o eroare în textul SCCIT
BIT3 —Ø— (BIT 10 inefectiv)
1— Inhibă tipărirea erorilor
BIT4 —Ø—
1— Ciclu pe subtestul definit de biţii 6,7
BIT5 —Ø—
1— Inhibă tipărirea listei de erori după fiecare execuţie a subtestului.
BIT6,7 — Cod subtest
BIT8 —Ø— Scrie la începutul fiecărui subtest numărul corespunzător.
—1 Oprire după fiecare subtest.
BIT9 —Ø
—1 Inhibă execuţia testului GALPAT
BIT10 —Ø
—1 Tipăreşte numai prima eroare din modul cînd BIT3=1.
BIT11 —Ø
—1 Execută subtestul PING PONG de 25 de ori
BIT12 —Ø
—1 Inhibă tipărirea lui, END PASS **'
BIT13 —Ø
—1 Inhibă ciclul pe execuţia testului RAM

Deci testul automat este caracterizat de următoarele :
— execuţia nu se opreşte la apariţia unei erori ;
— execută subtestele în ordinea crescătoare a codului ;
— tipăreşte erorile şi lista de erori ;
— inhibă erorile şi lista de erori ;
— inhibă subtestul GALPAT ;
— execută subtestul PING PONG o singură dată ;
— inhibă ciclul pe execuţia testului RAM

Cei 14 biţi se introduc după mesajul 'SWITCH :' (în regimul MANUAL) sub forma a 4 cifre hexa, astfel :

BIT15 BIT14 BIT13 BIT12 | BIT11 BIT1Ø BIT9 BIT8
cifră hexa | cifră hexa
SWITCH2

BIT7 BIT6 BIT5 BIT4 | BIT3 BIT2 BIT1 BITØ
cifră hexa | cifră hexa
SWITCH1

Ordinea de introducere este :

SWC2 (CR) SWC1 (CR).

Codificarea și ordinea de execuție în regim AUTOMAT sînt următoarele :

ØØ— Adresare (ADR)
Ø1— Scriere-citire (SCCIT)
Ø2— PING PONG
Ø3— GALPAT

14.5.1. Descrierea modurilor de lucru ale programului roșu

Testul adresare : (ADR) constă în scrierea succesivă în pagini de memorie de cîte 256 de octeți a unui contor cu valoarea Ø-FFH. La începutul fiecărei pagini contorul este dublu incrementat astfel încît se poate identifica ușor fiecare pagină de memorie. Informația înscrisă arată astfel :

```
CØØØ   ØØ  Ø1  Ø2 .... FF
C1ØØ   Ø1  Ø2  Ø3 .... FE
C2ØØ   Ø2  Ø3  Ø4 .... FD
  :
```

Testul se aplică pe zone compacte de memorie.

Facultativ se poate repeta testul, inversînd la scriere în memorie cei doi cuarteți ai contorului.

Testul scriere-citire (SCCIT) detectează biții de memorie blocați la zero/unu. Se scrie succesiv în fiecare locație configurația inițială (de exemplu AA), apoi complementul ei și din nou configurația inițială, verificîndu-se de fiecare dată corectitudinea operației. Se aplică pe zone compacte de memorie.

Testele PING PONG și GALPAT operează la nivel de modul de 16 Ko (Ø, 4ØØØH, 8ØØØH, CØØØH). Aceste teste necesită o subrutină (SGP) care delimitează din zona compactă de memorie RAM limitele modulului de testat (ADINF respectiv ADSUP).

Testul GALPAT : detectează defecte de adresare, selecție multiplă, timp de acces. După ce se scrie în modul configurația inițială, se parcurge succesiv fiecare locație : se scrie în ea configurația complementară și se verifică toate locațiile modulului. Fiecare verificare este urmată de o verificare a locației curente. La sfîrșit se reface locația curentă. Se repetă testul pentru configurația complementară. Întrucît operarea cu fiecare locație implică verificarea întregului modul, testul durează de ordinul a cîtorva ore.

Testul PING-PONG : este asemănător cu testul GALPAT, dar optimizat ca durată. După ce modulul de memorie este înscris cu configurația inițială se parcurge succesiv fiecare locație înscriindu-se cu configurația complementară. Pentru fiecare locație curentă se verifică 14 locații, fiecare verificare fiind urmată de verificarea locației curente. Adresele acestor locații se obțin pornind de la adresa locației curente, prin complementarea cîte unui bit de adresă.

În continuare se prezintă programul care execută acest test. Subrutina SGP apelată anterior a pregătit adresele limită (ADINF, ADSVP) pentru modulele testabile.

```
PING PONG: CAL INIT      ; iniţializare
P1:     CALL CYBR        ; copiază cuvînt iniţial între (ADINF, ADSUP)
        LDA ADINF
        MOV D,A          ; (D,E) — adresa cuvîntului test
        MVI E,ØØ
P2:     LDAX D
        XRA B            ; Se verifică dacă la adresarea cuvîntului de test s-a
        JZ P21           înscris configuraţia iniţială.
        PUSH B
        PUSH D
        CALL RWERR       ; subrutina de contorizare şi tratare a erorilor
        POP D
        POP B
P21:    MOV A,C          ; Se înscrie în locaţia test configuraţia complementară
        STAX D
        LDA ADINF
        CPI AØ
        JZ P7            ; Se stabileşte numărul de biţi care vor fi complemen-
        M VI H,15H       taţi din adresa cuvîntului test (pentru un modul întreg
        JMP P8           se complementează 14 biţi)
P7:     MVI H,14
P8:     MOV A,H
        SUI Ø9H
        STA PNG
        MVI L,8Ø         ; L — masca pentru complementarea unui bit din
        PUSH H.          adresă
        MOV H,D
        MOV L,E:         ; Adresa cuvîntului test în stivă iar în H şi L contorul
P3:     XT HL            şi masca pentru complementare.
        DCRH
        JZ P6
        MOV A,L
        RLC              ; masca devine L=01H
        MOV L,A
        LDA PNG
        CMP H
        MOV A,L
        HTHLL            ; Masca şi contorul în stivă
        JNC P4           ; salt dacă PNG≥contor de biţi
        XRA L
        MOV L,A          ; complementarea bitului din L
        JMP P5
P4:     XRA H            ; complementarea bitului din H
        MOV H,A
P5:     LDA ADMAX        ; Adresa sfîrşit RAM
        DCR A
        CMP H
        JC P9            ; salt dacă H≥HMAX.
        MOV A,M
        XRA B            ; locaţia derivată se compară cu configuraţia iniţială.
        JZ P9
        PUSH B
        PUSH D
        CALL RWERR.
```

```
        POP D
        POP B
P9 :    LDAX D       ; se compară cuvîntul test cu configuraţia complementată
        XRA C
        JZ P3        ; dacă nu este eroare salt la complementarea următorului
        PUSH B         bit de adresă
        PUSH D
        MOV B,C
        CALL RWERR
        POP D
        POP B
        JMP P3
P6      POP H        ; S-a terminat un cilu de verificare
        MOV A,B
        STAX D       ; Rescriere cuvînt iniţial la adresa
        INX D          cuvîntului de test
        LDA ADSUP
        CMP D
        JNZ P2       ; Dacă adresa următoare ≠ ADSUP se închide bucla
        MOV A,B        externă
        MOV B,C      ; schimbă cuvîntul iniţial cu cuplementarul său
        MOV C,A
        LDA VARPR1
        XRI 2Ø       ; Se testează şi modifică ITEST
        STA VARPR1
        ANI 2Ø       ; Dacă INTEST=1 se execută testul şi pentru configu-
        JNZ P1         raţia complementară
        RET
```

14.5.2. **Organizarea testului RAM.** În figura 14.2 este reprezentată schematic organigrama testului RAM. După iniţializări şi mesaje de început test, se cere precizarea octeţilor SWC2, SWC1 în regimul manual, iar în regimul automat ei sînt afişaţi, fiind determinaţi interior.

În regimul automat zonele de test RAM au fost precizate la intrarea în programul TEST aMIC. În funcţie de SWC4, se execută testele în ordinea crescătoare a codurilor sau se ciclează pe testul dat de SWC67.

Codul testului curent este memorat şi, pe baza lui, se crează adresa de salt la subtest şi codul eventualelor erori. Dacă SWC8=Ø atunci la începerea unui subtest se afişează codul subtestului curent, informînd prin aceasta că testul anterior s-a terminat (în acest regim programul nu se opreşte după subtest). După execuţia subtestului, dacă SWC8=1 se dă mesaj : END** şi se pune la dispoziţia operatorului. Comanda CTRL-G va executa listarea erorilor şi predă controlul monitorului TEST aMIC. Dacă SWC5=1 se îmbibă listarea erorilor şi continuă testul. In cazul în care subtestul terminat a fost GALPAT, se afişează sau nu END PASS**—funcţie de SWC12. Afişarea este utilă în cazul în care se ciclează pe testul RAM. Bitul SWC11 decide dacă subtestul PING PONG se face o singură dată sau de 25 de ori. Se observă din organigramă complexitatea regimurilor de lucru a programului TEST RAM.

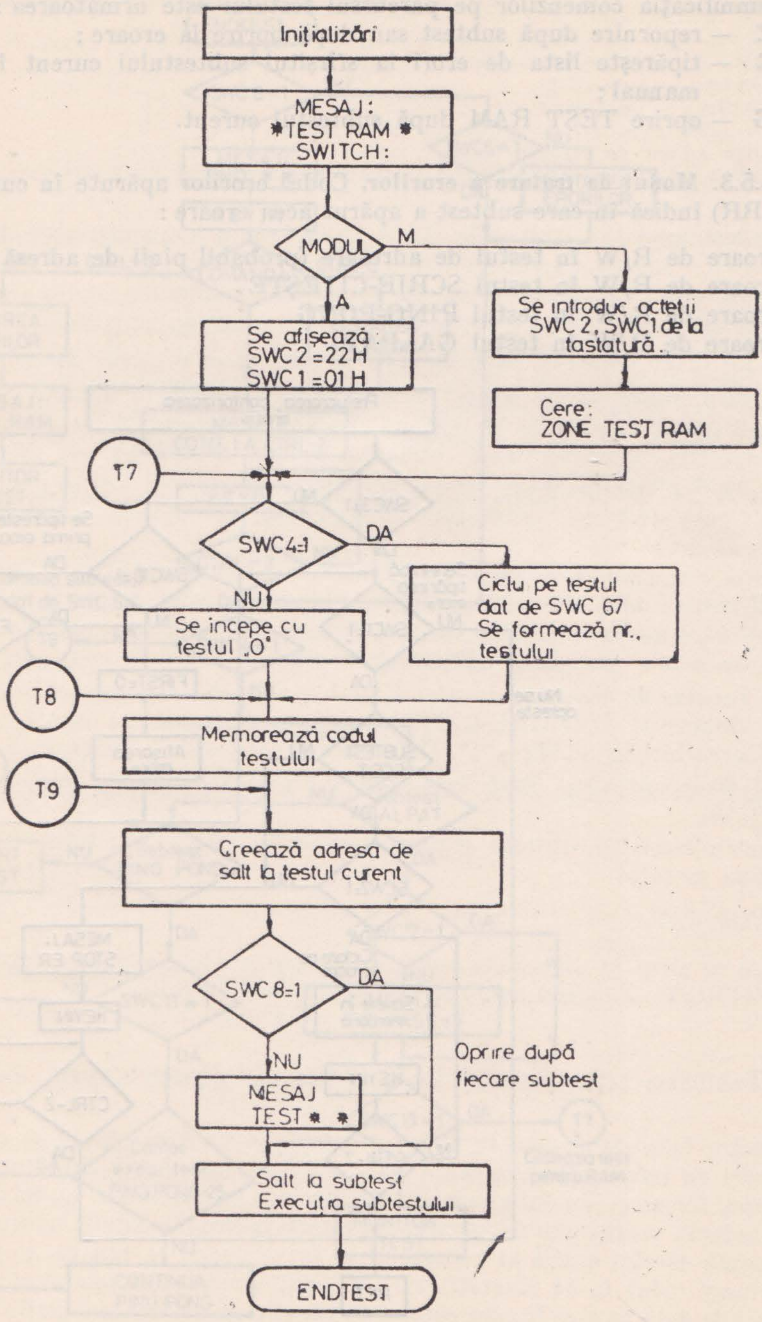


Fig. 14.2. Organigrama testului RAM.

Semnificaţia comenzilor pe parcursul testului este următoarea :

CTRL-Z — repornire după subtest sau după oprire la eroare ;

CTRL-C — tipăreşte lista de erori la sfîrşitul subtestului curent în regim manual ;

CTRL-G — oprire TEST RAM după subtestul curent.

14.5.3. Modul de tratare a erorilor. Codul erorilor apărute în cursul testelor (ERR) indică în care subtest a apărut acea eroare :

Ø1 — Eroare de R/W în testul de adresare (probabil pinii de adresă defecţi)
Ø2 — Eroare de R/W în testul SCRIE-CITEŞTE
Ø3 — Eroare de R/W în testul PING-PONG
Ø4 — Eroare de R/W în testul GALPAT

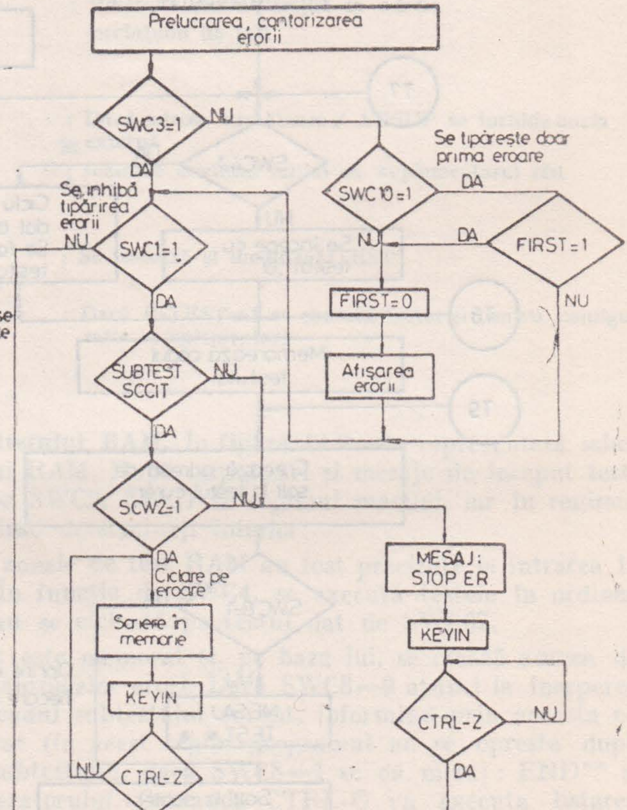
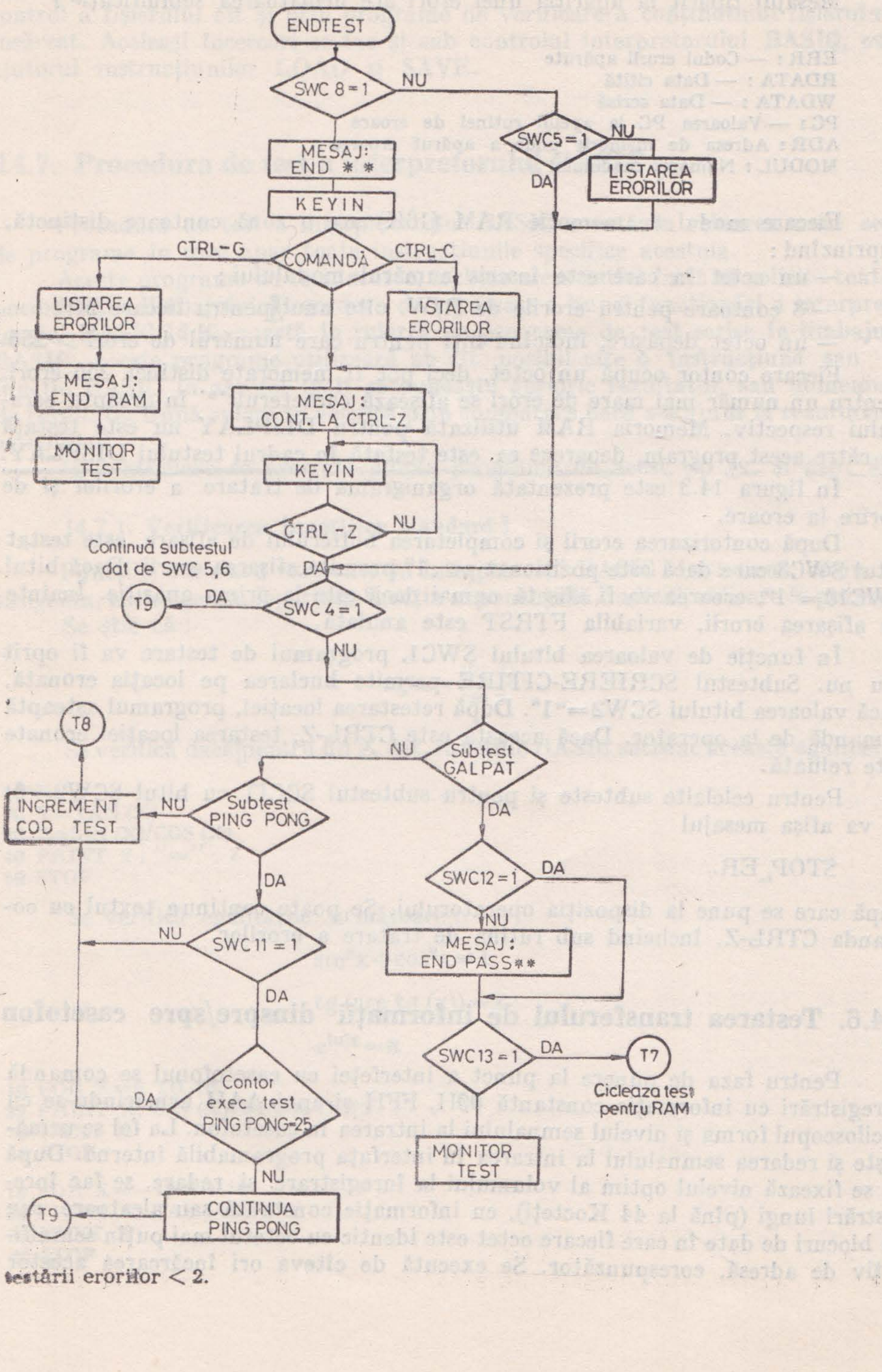


Fig. 14.3. Organigrama



testării erorilor < 2.

Mesajul tipărit la apariţia unei erori are următoarea semnificaţie:

ERR : — Codul erorii apărute
RDATA : — Data citită
WDATA : — Data scrisă
PC : — Valoarea PC la apelul rutinei de eroare
ADR : Adresa de memorie unde a apărut eroarea
MODUL : Numărul modulului

Fiecare modul de memorie RAM (16K) are o zonă contoare distinctă, cuprinzînd:

— un octet în care este înscris numărul modulului;
— 8 contoare pentru erorile de R/W, cîte unul pentru fiecare bit;
— un octet depăşire, indicînd biţi pentru care numărul de erori > 256.

Fiecare contor ocupă un octet, deci pot fi memorate distinct 256 erori. Pentru un număr mai mare de erori se afişează caracterul '*' în dreapta ecranului respectiv. Memoria RAM utilizată pentru DISPLAY nu este testată de către acest program, deoarece ea este testată în cadrul testului DISPLAY.

În figura 14.3 este prezentată organigrama de tratare a erorilor şi de oprire la eroare.

După contorizarea erorii şi completarea bufferului de afişare, este testat bitul SWC3 care dacă este poziţionat pe „1" permite afişarea erorii. Dacă bitul SWC1Ø="1", eroarea va fi afişată numai dacă este la prima apariţie. Înainte de afişarea erorii, variabila FTRST este anulată.

În funcţie de valoarea bitului SWC1, programul de testare va fi oprit sau nu. Subtestul SCRIERE-CITIRE permite buclarea pe locaţia eronată, dacă valoarea bitului SCW2="1". După retestarea locaţiei, programul aşteaptă comandă de la operator. Dacă aceasta este CTRL-Z, testarea locaţiei eronate este reluată.

Pentru celelalte subteste şi pentru subtestul SCCIT cu bitul SCW2=Ø, se va afişa mesajul

STOP ER.

după care se pune la dispoziţia operatorului. Se poate continua textul cu comanda CTRL-Z, încheind sub rutina de tratare a erorilor.

14.6. Testarea transferului de informaţii dinspre/spre casetofon

Pentru faza de punere la punct a interfeţei cu casetofonul se comandă înregistrări cu informaţie constantă ØØH, FFH şi apoi AAH urmărindu-se cu osciloscopul forma şi nivelul semnalului la intrarea în casetofon. La fel se urmăreşte şi redarea semnalului la intrarea în interfaţa programabilă internă. După ce se fixează nivelul optim al volumului la înregistrare şi redare, se fac înregistrări lungi (pînă la 44 Kocteţi), cu informaţie constantă sau aleatoare, sau cu blocuri de date în care fiecare octet este identic cu octetul mai puţin semnificativ de adresă, corespunzător. Se execută de cîteva ori încărcarea acestor

fişiere în memoria calculatorului şi se verifică încărcarea atît prin suma de control a fişierului cît şi prin programe de verificare a conţinutului fişierului încărcat. Aceleaşi încercări se fac şi sub controlul interpretorului BASIC, cu ajutorul instrucţiunilor LOAD şi SAVE.

14.7. Procedura de test a interpretorului BASIC

Procedura de test a interpretorului BASIC constă în reluarea unui set de programe în care apar toate instrucţiunile specifice acestuia.

Aceste programe de test, scurte, trebuie alese astfel încît să solicite toate posibilităţile limbajului. Procedura de verificare a bunei funcţionări a interpretorului BASIC 14 K, constă în rularea de programe de test scrise în limbajul BASIC. Aceste programe utilizează pe cît posibil cîte o instrucţiune sau o funcţie BASIC în aşa fel încît se cunoaşte aprioric rezultatul, sau domeniul de rezultate. După rulare se verifică prin comparare dacă s-a ajuns la rezultatul aşteptat.

În continuare se prezintă cîteva programe de acest tip (v. şi cap. 9).

14.7.1. Verificarea funcţiilor standard

Funcţiile standard rezolvate în interpretorul BASIC sînt: sinus, cosinus, tangenta, arctangenta, logaritm, funcţia exponenţială, radical, ridicare la putere.

Se ştie că:

$$\text{tg}\, x = \frac{\sin x}{\cos x}$$

Se verifică dacă pentru un X dat, funcţiile BASIC satisfac această egalitate.

```
1Ø X=Ø
2Ø Y=TAN (X)
3Ø Z=SIN (X)/COS (X)
4Ø PRINT Y ; "=" ; Z
5Ø STOP
```

Se verifică egalităţile următoare:

$$\sin^2 x + \cos^2 x = 1$$

$$\text{tg}\,(\text{arc tg}\,(x)) = x$$

$$e^{\ln x} = x$$

```
1Ø FOR X=Ø TO 1Ø
2Ø PRINT X, SIN (X)↑2+COS (X)↑2
3Ø NEXT X
4Ø STOP
```

```
1Ø FOR X=—1Ø TO 1Ø
2Ø PRINT X, TAN (ATN (X))
3Ø NEXT X
4Ø STOP
```

```
10 FOR X=1 TO 10
20 PRINT X, EXP(LOG(X))
30 NEXT X
40 STOP
```

Se calculează rădăcina pătrată dintr-un număr dat de utilizator prin formula interactivă :

$$y_{i+1}=\frac{1}{2}\left(y_i+\frac{A}{y_i}\right)$$

unde A este numărul introdus ;

y_i este o valoare arbitrară iniţială

Iteraţia se opreşte cînd $(y_{i+1}-y_i)=\emptyset$. Se tipăreşte valoarea astfel obţinută. Se tipăreşte apoi şi valoarea obţinută prin funcţia BASIC de rădăcină pătrată (SQR) şi se compară cele două rezultate astfel obţinute.

```
10 REM "RADACINA PATRATA"
20 PRINT "DATI NUMARUL"
30 INPUT A
40 B=INT (RND (X)*A)+1
50 REM "B=Y ARBITRAR ALES"
55 REM "IN FUNCTIE DE A"
60 Y=0.5*(B+A/B)
70 IF B=Y THEN 100
80 B=Y
90 GOTO 60
100 PRINT "RADACINA PATRATA" ;
105 PRINT "PRIN ITERATIE :" ; Y
110 X=SQR (A)
120 PRINT "RADACINA PATRATA" ;
125 PRINT "PRIN FUNCTIE SQR : " ; X
130 Z=ABS (X—Y)
140 PRINT "DIFERENTA :" ; Z
150 STOP
```

Funcţia internă de generare numere aleatoare (RANDOM), pune la dispoziţia utilizatorului cîte un număr aleator în domeniul (0, 1). Se generează tabele de cîte 10 numere variabile în diferite domenii, înmulţind numărul aleator cu diferite constante şi/sau luînd valoarea întreagă cu funcţia INTEGER.

```
10 REM "VERIFICARE  R A N D O M"
20 PRINT "VALORI REALE ÎNTRE 0 SI 1"
30 FOR I=1 TO 10
40 PRINT RND (X),
50 NEXT I
60 PRINT "VALORI REALE ÎNTRE 0, 10 SI 0, 100"
70 FOR I=|1|TO 10
80 PRINT RND (X)*10 ; RND (X)*100
90 NEXT I
100 PRINT "VALORI INTREGI INTRE 0, 10 SI 0, 100"
110 FOR I=1 TO 10
120 PRINT INT (RND (X)*10) ; INT (RND (X)*100)
130 NEXT I
```

```
14Ø PRINT "VALORI INTREGI INTRE Ø, 6 ŞI 1Ø, 2Ø"
15Ø FOR I=1 TO 1Ø
16Ø PRINT INT (RND (X)*6) ; INT (RND (X) * 1Ø+1Ø)
17Ø NEXT I
18Ø STOP
```

Prin funcţia PLOT se poate aprinde un punct pe ecranul TV, considerînd ecranul de 64*64 puncte. Se trasează curba sinus cu funcţia PLOT şi se şterge cu funcţia UNPLOT.

```
1Ø INIT P
2Ø FOR N=1 TO 64
3Ø PLOT N, 22+2Ø*SIN (N/32*PI)
4Ø NEXT N
5Ø FOR N=1 TO 64
6Ø UNPLOT N, 22+2Ø*SIN (N/32*PI)
7Ø NEXT N
8Ø STOP
```

Funcţia RANDOM se poate verifica şi cu funcţia PLOT. Se umple ecranul aleator prin funcţiile RANDOM şi PLOT cu puncte aprinse. Dacă această variaţie aleatoare umple relativ uniform ecranul, funcţia RANDOM este aleatoare.

```
1Ø INIT P
2Ø PLOT INT (RND (X)*64)+1, INT (RND (X)*64+1)
3Ø GOTO 2Ø
4Ø END
```

Funcţia PLOT aprinde un punct pe ecran şi UNPLOT stinge punctul pe ecran. Se poate astfel aprinde şi stinge acelaşi punct obţinînd astfel o pulsaţie a unui punct.

```
10 INIT P
2Ø X=RND (X)
3Ø Y=RND (Y)
40 FOR I=1 TO 1Ø
5Ø PLOT INT (X*64)+1, INT (Y*64)+1
60 UNPLOT INT (X*64)+1, INT (Y*64)+1
70 NEXT I
80 STOP
```

Funcţia PRINT AT (X, Y) permite afişarea pe ecran, la coordonate date, a unei informaţii. Ecranul este văzut ca formînd 32 de linii a cîte 3Ø de caractere. În instrucţiune, X este linia şi Y coloana de unde se va începe afişarea.

Se afişează pe ecran un tabel de formă cunoscută.

```
1Ø INIT P
2Ø PRINT AT (2, 2) ; "TABEL CU NUMERE" ;
25 PRINT "DE TELEFOANE"
3Ø PRINT AT (3, 2) ; S$
```

```
40 PRINT AT (6, 5) ; "NUME I PREFIX I" ;
45 PRINT "NR. TEL."
50 PRINT AT (7, 2) ; S$
60 FOR I=8 TO 16 STEP 2
70 READ N $ (10)
80 READ N
100 PRINT AT (I, 2) ; N $; "I" ; P ; "I" ; N
110 PRINT AT (I+1, 2) ; S$
120 NEXT I
130 STOP
140 DATA "NUME 1", 971, 12345
150 DATA "NUME 2", 90, 457321
160 DATA ....
.
.
.
200 DATA ...
210 S$="————————————————————————————————"
220 END
```

Aprinderea şi stingerea unei zone prin instrucţia PRINT AT (X, Y)

```
10 INIT P
20 X=INT (RND (X)*64)+1
30 Y=INT (RND (Y)*64)+1
40 FOR I=1 TO 20
50 PRINT AT (X, Y) ; " . "
60 PRINT AT (X, Y); "  "
70 NEXT I
80 STOP
```

Caracterul care aparţine la o valoare A se obţine prin CHR$ (A). Setul de caractere se afişează prin următorul program.

```
10 PRINT "SETUL DE CARACTERE"
20 FOR A=0 TO 256
30 PRINT A, CHR$ (A)
40 NEXT A
50 STOP
```

Funcţia INKEY$ permite utilizatorului să dea unei variabile şir valoarea tastei apăsate. Următorul program permite utilizarea aMIC-ului ca maşină de scris. Apăsarea tastei RETURN poziţionează pe linie nouă.

```
10 INITP
20 A$=INKEY$
30 B$=INKEY$
40 IF A$=B$ THEN 30
50 REM "S-A TASTAT UN CARACTER"
60 PRINT B$
70 GOTO 20
80 END
```

Valoarea variabilei şir se obţine prin funcţia VAL.

Dacă la linia 2Ø schimbăm expresia şi la linia 3Ø valoarea lui X, putem efectua verificări pentru diferite expresii şi valori ale lui X.

```
1Ø PRINT "VALOAREA EXPRESIEI"
2Ø F$ ="2X ↑ 2—X+1"
3Ø FOR X=1 TO 5
4Ø T=VAL (F$ )
5Ø PRINT "EXPRESIA :" ; F$.
6Ø PRINT "PENTRU X=" ; X ;
7Ø PRINT "ARE VALOAREA :" ; T
8Ø NEXT X
9Ø STOP
```

Se permit în Interpretorul BASIC trei bucle suprapuse. În următorul program există trei bucle K, J şi I. Totodată se lucrează cu instrucţiuni matriciale.

```
1Ø PRINT "ORDONAREA UNUI SIR"
2Ø READ N
3Ø DIM A (N)
4Ø MAT READ A
5Ø FOR I=1 TO N—1
6Ø FOR J=I+1 TO N
7Ø IF A (I)<A (J) THEN 13Ø
8Ø B=A (J)
9Ø FOR K=J TO I+1 STEP —1
1ØØ A (K)=A (K—1)
11Ø NEXT K
12Ø A (I)=B
13Ø NEXT J
14Ø NEXT I
15Ø PRINT "ŞIRUL ORDONAT"
16Ø MAT PRINT A
17Ø STOP
18Ø DATA 5
19Ø DATA 1Ø, 3Ø, 5, 15, 5
2ØØ END
```

În funcţie de valoarea unui număr se pot executa diferite secvenţe de program cu instrucţiunea ON. Pentru a verifica această instrucţiune se tastează un număr X de utilizator şi dacă pe ecran apare „aţi bătut X" atunci funcţionează corect.

```
1Ø REM "VERIFICARE ON — GOTO"
2Ø PRINT "TASTATI UN NUMĂR"
3Ø INPUT M
4Ø ON M GOTO 8Ø, 1ØØ, 12Ø, 14Ø, 16Ø, 18Ø, 2ØØ, 22Ø, 24Ø
5Ø PRINT "ATI TASTAT 0"
6Ø GOTO 2Ø
8Ø PRINT "ATI TASTAT 1"
9Ø GOTO 2Ø
1ØØ PRINT "ATI TASTAT 2"
11Ø GOTO 2Ø
12Ø PRINT "ATI TASTAT 3"
13Ø GOTO 2Ø
```

```
140 PRINT "ATI TASTAT 4"
150 GOTO 20
160 PRINT "ATI TASTAT 5"
170 GOTO 20
180 PRINT "ATI TASTAT 6"
190 GOTO 20
200 PRINT "ATI TASTAT 7"
210 GOTO 20
220 PRINT "ATI TASTAT 8"
230 GOTO 20
240 PRINT "ATI TASTAT 9"
250 GOTO 20
260 END
```

Utilizarea instrucţiunilor READ, MATREAD, RESTORE se exemplifică în următorul program. A se observa că nu se ajunge niciodată la citirea valorilor 3 din blocul de date.

```
10 REM "DATA, READ, MAT READ"
20 READ A, B, C
30 PRINT A, B, C
40 READ A, B, C
50 PRINT A, B, C
60 RESTORE
70 MAT READ D (6)
80 MAT PRINT D
90 STOP
100 DATA 1, 1, 1, 2, 2, 2, 3, 3, 3
110 END
```

Următorul program utilizează instrucţiunile cu şiruri de caractere.

```
10 PRINT "SUPRAFAŢA"
15 PRINT "DREPTUNGHI, TRIUNGHI, CERC"
20 INPUT S$
30 IF S$ ="DREPTUNGHI" THEN 70
40 IF S$ ="TRIUNGHI" THEN 110
50 IF S$ ="CERC" THEN 160
60 GOTO 20
70 PRINT "DATI LUNGIMEA SI LATIMEA"
80 INPUT B, C
90 Z=B*C
100 GOTO 190
110 PRINT "DATI LUNGIMILE LATURILOR"
120 INPUT A, B, C
130 S=(A+B+C)/2
140 Z=SQR (S* (S—A)*(S—B)*(S—C))
150 GOTO 190
160 PRINT "DATI RAZA"
170 INPUT R
180 Z=PI*R ↑ 2
190 PRINT "SUPRAFATA" ; S$ ; "="; Z
200 GOTO 10
210 END
```

Apelul de subprogram scris în limbaj BASIC se face prin GOSUB nr. Prima instrucţiune RETURN întîlnită provoacă revenirea la instrucţiunea

următoare instrucţiunii GOSUB executate ultima dată. În următorul program se verifică unele posibilităţi de salturi la subprograme în BASIC. Primul RETURN întîlnit la care nu există pereche GOSUB provoacă oprirea executării în continuare a programului.

```
20 REM "UTILIZĂRI GOSUB ŞI RETURN"
25 REM "APEL SPR. LA PRIMA INTRARE"
30 GOSUB 200
35 PRINT "REVINE DIN SPR 200"
40 REM "APEL SPR. INTERIOR"
50 GOSUB 210
60 PRINT "REVINE DIN SPR 210"
55 REM "APEL SPR. 300 REVINE CU GOTO"
60 GOSUB 300
65 PRINT "REVINE AICI PRIN RETURN URMĂTOR"
70 PRINT "REVINE PRIN GOTO DIN SPR 300"
75 REM "URM. RETURN E PRIMUL ÎNTÎLNIT"
80 REM "DUPĂ GOSUB 300, REVINE DUPĂ EL"
85 RETURN
90 REM "AICI NU SE AJUNGE"
95 STOP
200 PRINT "INTRARE SPR. 200"
210 PRINT "CORPUL SPR. 200"
220 RETURN
300 PRINT "INTRARE SPR. 300"
310 GOTO 70
400 END
```

```
Pe ecran apare deci :
Intrare spr. 200
Corpul spr. 200
Revine din spr. 200
Corpul spr. 200
Revine din spr. 210
Intrare spr. 300
Revine prin GOTO din spr. 300
Revine aici prin RETURN următor.
```

Utilizarea subprogramelor scrise în limbaj maşină e posibilă prin instrucţiunea CALL. Următorul program în limbaj BASIC cheamă primul subprogram în cod maşină

```
10 REM "CALL"
20 CALL (1)
30 STOP
```

Subprogramul scris în limbaj maşină se introduce în memoria calculatorului decodificat în hexazecimal, prin funcţia S (substituţie) din monitor.

Se doreşte introducerea unui subprogram care să genereze în mijlocul ecranului caracterul A, diferit de caracterul A dat de generatorul de caractere al monitorului. Se cheamă subprogramul de iniţializare ecran al monitorului şi subprogramul de scriere caracter.

```
CALL INITV
LXI H, (ADR MIJLOC ECRAN TV)
```

```
        LXI D, (ADR GENERATOR A PROPRIU)
        MVI A, 6
        CALL WR26
        RET
CARGEN  DB 6, 9, 9, F, 9, 9,
```

Se va folosi următoarea secvenţă de substituire S:

```
6400    Ø1           ; NR. SUBPROGRAM
        FF 66        ; ADR SPR 1
        FF           ; SFIRSIT TABEL SPR
66FF    CD A4 Ø2     ; CALL INITV
        21 ØA 5Ø     ; LX1 H, 5ØØA
        11 OE 67     ; LXI D, CARGEN
        3E Ø6        ; MVI A, 6
        CD Ø5 Ø2     ; CALL WR26
        C9           ; RET
67OE    Ø6 Ø9 Ø9     ; CARGEN PT. A
        ØF Ø9 Ø9
```

Pentru verificarea funcţiilor grafice MOVE, DRAW se dau în continuare cîteva curbe specifice:

a) O serie de curbe sinusoidale

```
2Ø INIT P
3Ø X=Ø
4Ø Y=5Ø
5Ø Z=X
6Ø MOVE X, Y
7Ø FOR I=Ø TO 3*PI STEP PI/1Ø
8Ø DRAW Z, SIN (I)*25+Y
9Ø Z=Z+2
1ØØ NEXT 1
11Ø X=X+2
12Ø Y=Y+2
13Ø IF X <=26 THEN 5Ø
14Ø STOP
```

b) O serie de cercuri care formează un con. A, B — coordonatele centrului primului cerc.

```
2Ø READ A, B
3Ø READ X, Y
4Ø INIT P
5Ø FOR T=1 TO 25
6Ø MOVE A+T, B
7Ø FOR I=1 TO 2*PI STEP PI/1Ø
8Ø DRAW A+T*COS (I), B+T*SIN (I)
9Ø NEXT I
1ØØ A=A+X
11Ø B=B+Y
12Ø NEXT T
13Ø STOP
14Ø DATA 1Ø, 1Ø, 5, 5
15Ø END
```

c) Linii de lungimi variabile din cele patru colţuri ale ecranului

```
10 INIT P
20 MOVE 0, 0
30 DRAW RND (X)*100, RND (X)*100
40 MOVE 100, 100
50 DRAW 100—RND (X)*100, 100—RND (X)*100
60 MOVE 0, 100
70 DRAW RND (X)*100, 100—RND (X)*100
80 MOVE 100, 0
90 DRAW 100—RND (X)*100, RND (X)*100
100 GOTO 20
110 END
```

Trasarea unei figuri cu instrucţiile BASIC RMOVE şi RDRAW şi rotirea figurii cu ROTATE. Se trasează un triunghi şi se roteşte acest triunghi.

```
10 INIT P
20 WINDOW —50, 50, —50, 50
25 MOVE 0, 0
30 FOR U=0 TO 2*PI STEP PI/10
35 ROTATE U
40 RDRAW 10, 0
45 RDRAW 10, 10
50 RDRAW —20, —10
55 NEXT U
60 STOP
```

Verificarea funcţiei WINDOW şi VIEWPORT. Se trasează un cerc de o mărime dată la diferite valori pentru WINDOW şi VIEWPORT. Se observă că trebuie aleasă mărimea şi zona ecranului în funcţie de mărimea figurii.

```
10 INIT P
20 REM "WINDOW SI VIEWPORT IMPLICIT"
25 REM "0, 100, 0, 100"
30 REM "APARE PE ECRAN 1/4 CERC"
40 GOSUB 200
50 INIT P
60 WINDOW —50, 50, —50, 50
70 REM "VIEWPORT IMPLICIT"
75 REM "0, 100, 0, 100
80 REM "APARE TOT CERCUL ÎN MIJLOC ECRAN"
90 GOSUB 200
100 INIT P
110 WINDOW —50, 50, —50, 50
120 VIEWPORT 50, 100, 50, 100
130 REM "APARE CERCUL ÎN DREAPTA SUS"
140 GOSUB 200
150 STOP
200 MOVE 25, 0
210 FOR I=0 TO 2*PI STEP PI/10
220 DRAW 25*COS (I), 25*SIN (I)
230 NEXT I
240 FOR I=1 TO 50
250 REM "TEMPORIZARE"
```

```
260 NEXT I
270 RETURN
280 END
```

Trasări de pătrate care se măresc ; în video normal şi video invers.

```
20 INIT P
30 PUT (34)=128
40 GOSUB 100
50 INIT P
60 PUT (34)=160
70 GOSUB 100
80 GOTO 20
100 MOVE 50, 50
110 FOR L=10 TO 100 STEP 10
120 MOVE 50—L/2, 50—L/2
130 RDRAW L, 0
140 RDRAW 0, L
150 RDRAW —L, 0
160 RDRAW 0, —L
170 NEXT L
180 RETURN
190 END
```

Bateria de teste, odată trecută, ne dă posibilitatea de a opera cu încredere cu exemplarul de aMIG.

ANEXA 3

COLECŢIE DE PROGRAME PENTRU REZOLVAREA UNOR PROBLEME DE MATEMATICĂ DIN MATERIA CLASELOR A IX-A ŞI A X-A

Proiectarea şi realizarea microcalculatoarelor individuale în ţara noastră creează premisele necesare introducerii lor în învăţămîntul liceal, în cadrul laboratoarelor de matematică, fizică, biologie, etc.

Trebuie amintit faptul că în acest an a intrat în fabricaţie de serie, la Întreprinderea de memorii electronice din Timişoara, microcalculatorul aMIC.

Ministerul Educaţiei şi Învăţămîntului a comandat circa o sută exemplare, care vor intra în acest an în dotarea unor instituţii de învăţămînt superior. Se prevede ca acţiunea de dotare să se extindă şi la nivelul liceelor.

Între timp, specialiştii în tehnica de calcul din ţara noastră au proiectat noi tipuri de microcalculatoare individuale, dintre care se amintesc : HC-85, PRAE şi DEGA-209.

Interesant este microsistemul HC-85, care a fost proiectat la Catedra de calculatoare din Institutul Politehnic Bucureşti şi care va intra în curînd în fabricaţie de serie la Întreprinderea de Calculatoare Electronice.

Avînd în vedere performanţele acestor calculatoare legate, de implementarea unor limbaje evoluate (BASIC, PASCAL, FORTRAN, MICROPROLOG, FORTH, etc.), posibilitatea de folosire a unui ecran color pentru vizualizare, cît şi faptul că ele vor putea intra în dotarea liceelor, în cele ce urmează se va prezenta o colecţie de programe în BASIC, pentru asistarea studiului unor capitole de matematică din materia claselor a IX-a şi a X-a.

1. PROGRAME PENTRU CALCULE CU POLINOAME

Se prezintă programe pentru :

— împărţirea unui polinom de un grad oarecare n cu polinoame de gradul întîi şi doi, ireductibile şi unitare în R, adică de forma ; x+ a şi x ↑ 2+ px+ q,

— calculul valorii unui polinom,

— calculul rădăcinii reale a unui polinom într-un interval dat, ştiind că la capetele intervalului $[a, b]$, polinomul ia valori de semne contrare şi că în acest interval funcţia polinomială este strict monotonă.

1.1. PROGRAM PENTRU ÎMPĂRŢIREA UNUI POLINOM CU UN BINOM

Fie polinomul :

$$\sum_{i=0}^{n} a_i x^{n-i} \text{ şi binomul : } x+a.$$

Considerînd cîtul de forma : $\sum_{i=1}^{n} b_i x^{n-i}$ şi restul de forma : b_{n+1} rezultă următoarele :

$$\sum_{i=0}^{n} a_i x^{n-i} = \left(\sum_{i=0}^{n} b_i x^{n-i}\right)\cdot(x+a)+b_{n+1} \quad \text{sau}$$

$$\sum_{i=0}^{n} a_i x^{n-1} = b_0 x^{n+1} + \sum_{i=0}^{n} (b_{i+1}+ab_i)\cdot x^{n-i}.$$

Prin identificare se obţin următoarele relaţii :

$b_0=0$

$a_i=b_{i+1}+a\cdot b_i$, pentru $i=0, 1, 2, \ldots, n$

sau :

$b_{i+1}=a_i-a\cdot b_i$

şi restul :

$b_{i+1}=a_i-a\cdot b_i$

```
20 PRINT "introduceţi gradul n al polinomului"
30 INPUT n
40 PRINT "n=" ; n
50 PRINT "introduceţi coeficienţii polinomului SUMA"
(a(i)*x^(n-1))
60 DIM a (2*n)
65 DIM b (2*n)
70 FOR i=0 to n
80 PRINT "a (" ; i ;")="
85 INPUT a (i+1)
90 PRINT a (i+1)
100 NEXT i
110 PRINT "introduceţi termenul liber a al divizorului"
120 INPUT a
130 PRINT "a=" ; a
200 REM calculul citului
210 b (1)=0
220 FOR I=1 TO n+1
230 LET b (i+1)=a (i)-a*b (i)
240 NEXT i
300 REM afisare rezultate
310 FOR i=1 TO n
320 PRINT "b (" ; i ;")=" ; b" i+1)
330 NEXT i
340 PRINT "r=" ; b (n+2)
350 END
```

1.2. PROGRAM PENTRU ÎMPĂRŢIREA UNUI POLINOM DE GRADUL n CU TRINOMUL × ↑ 2+ px+ q.

Se consideră :

— polinomul de forma : $\sum_{i=0}^{n} a_i x^{n-i}$

— citul de forma : $\sum_{i=0}^{n} b_i x^{n-i}$

— restul de forma $rx+s$,

prin identificare, din relaţia :

$$\sum_{i=0}^{n} a_i x^{n-i} = \left(\sum_{i=0}^{n} b_i x^{n-i} \right) \cdot (x\uparrow 2+px+q)+rx+s$$

se obţin următoarele rezultate :

$b_0=0$ $r=b_{n+1}$

$b_1=1$ $s=b_{n+2}+pb_{n+1}$

$b_{i+2}=a_i-pb_{i+1}-qb_i$

```
10 REM împărţirea unui polinom de gradul n cu un trinom
20 PRINT "introduceţi gradul n al polinomului"
30 INPUT n
40 PRINT "n=" ; n
50 PRINT "introduceţi coeficienţii polinomului SUMA (a (i) * x^(n-1))"
60 DIMA (2*n)
65 DIMB (2*n)
70 FOR i=0 TO n
80 INPUT a (i+1)
90 PRINT "a (" ; i ; "0)=" ; a (i+1)
100 NEXT i
200 PRINT "introduceţi coeficienţii divizorului x^2+p.x+q"
210 PRINT "introduceţi p"
220 INPUT P
225 PRINT "p=" ; P
230 PRINT "introduceţi q"
240 INPUT q
250 PRINT "q=" ; q
300 REM calculul coeficienţilor citului
```

```
310 LET b (1)=0
320 LET b (2)=0
330 FOR i=1 TO n+1
340 LET b (i+2)=a (i)-p*b (i+1)-q*b (i)
350 NEXT i
400 REM afişare rezultate
410 FOR i=2 TO n
420 PRINT "b ("; i; ")="; b (i+1)
430 NEXT i
440 PRINT "r="; b (n+2)
450 PRINT "s="; b (n+3)+p*b (n+2)
500 END
```

1.3. PROGRAM PENTRU CALCULUL VALORII UNUI POLINOM DE GRADUL n

Programul foloseşte algoritmul lui Horner pentru calculul valorii polinomului:

$$P(x)+\sum_{i=0}^{n} a_i x^{n-i}$$

Polinomul P (x) poate fi scris şi sub următoarea formă:

$$P(x)=a_n+x(\ldots+x(a_2+x(a_1+x(a_0)))\ldots\ldots)$$

Calculul începe cu produsul cu cel mai mare grad de imbricare x·a.

```
10 REM calculul valorii unui polinom de gradul n
20 PRINT" introduceţi gradul n al polinomului"
30 INIT n
40 PRINT "n="; n
50 PRINT "introduceţi coeficienţii polinomului SUMA (a(i)*x^(n-i))"
60 DIM a (2*n)
70 FOR i=0 TO n
80 INPUT a (i+1)
90 PRINT "a ("; i; ")="; a (i+1)
100 NEXT i
110 PRINT "introduceţi valoarea lui x"
120 INPUT x
130 PRINT "x="; x
200 REM calculul valorii polinomului prin metoda Horner
210 LET p=a (1)
220 FOR i=1 TO n
230 LET p=p*x+a (i+1)
240 NEXT i
300 REM afişarea rezultatului
310 PRINT "p ("; x; ")="; p
320 END
```

1.4. STABILIREA SOLUŢIEI UNEI ECUAŢII PRIN METODA DIHOTOMIEI

Se ştie că dacă funcţia f este continuă şi strict monotonă în intervalul [a, b] şi dacă $f(a)\cdot f(b)<0$, atunci ecuaţia: $f(x)=0$ are o rădăcină unică în [a, b].

Dacă e este un număr pozitiv şi dacă $x_0 \in [a, b]$, iar $f(x_0-e)\cdot f(x_0+e)<0$, atunci x_0 este o rădăcină a ecuaţiei date cu o precizie e.

Calculul numeric al produsului $f(x_0-e)\cdot f(x_0+e)$ este în general aproximativ. Se impune ca precizia să fie suficientă pentru ca

$$f(x_0-e)\cdot f(x_0+e)<0.$$

Se consideră jumătatea intervalului [a, b] : $c=(a+b)/2$ şi se testează semnul lui $f(a)\cdot f(c)$. Dacă $f(a)\cdot f(c)<0$, rădăcina aparţine lui [a, c], în caz contrar aparţine lui [c, b].

Se construiesc următoarele intervale:

$[a_0, b_0]=[a, b]$

$$[a_1, b_1]=\begin{cases}[a, c], \text{ dacă } f(a)\cdot f(c)<0\\ [c, b], \text{ dacă } f(a)\cdot f(c)>0\end{cases}$$

....

$[a_n, b_n]=$ ....

Lungimea intervalului $[a_n, b_n]$ este (b−a)/2 ↑ n. Dacă se doreşte o precizie e, atunci este suficient să se construiască $[a_n, b_n]$ pentru un n astfel încît (b−a)/2 ↑ n<e.

```
10 REM stabilirea soluţiei unei ecuaţii folosind metoda dihotomiei
20 PRINT "introduceţi datele : a=lim. stînga ; b=lim. dreapta ; e=precizia"
30 INPUT a
35 PRINT "a=" ; a
40 INPUT b
45 PRINT "b=" ; b
40 INPUT e
50 PRINT "e=" ; e
51 PRINT "introduceţi gradul n al polinomului"
52 INPUT n
56 PRINT "n=" ; n
58 PRINT "introduceţi coeficienţii"
60 DIM a (2*n)
62 FOR i=0 TO n
64 INPUT a (i+1)
66 PRINT "a (" ; i ; ")= "; a (i+1)
68 NEXT i
80 REM calculul lui f (a) si f (b)
90 LET c=a
92 GO SUB 1210
94 LET x=z
96 LET c=b
98 GO SUB 1210
99 LET y=z
100 LET c=(a+b)/2
110 GO SUB 1210
120 LET d=x*z
130 IF d=0 THEN 200
140 IF d < 0 THEN 160
150 LET a=c
153 LET x=z
155 GO TO 170
160 LET b=c
165 LET y=z
170 IF b−a > e THEN 100
200 PRINT "c=" ; c
210 STOP
1210 LET p=a (1)
1220 FOR i=1 TO n
1230 LET p=p*c+a (i+1)
1240 NEXT i
1250 LET z=p
1260 RETURN
```

2. CALCULUL INTEGRALEI DEFINITE PRIN METODA SIMPSON

Pentru calculul integralei definite se foloseşte formula :

$$S=\int_{x_0}^{x_{2p}} f(x)\cdot dx=(h/3)\cdot[(y_0+y_{2p})+4(y_1+y_3+\ldots+y_{2p-1})]+2(y_2+y_4+\ldots+y_{2p-2})$$

unde :

h = $(x_{2p}-x_0)/2\cdot p$ este pasul,
p = numărul de diviziuni.
y = f (x) se dă la linia 500 din program ; în cazul de faţă
y= ((x−2)· x−1)*x+2.

```
10 REM CALCULUL INTEGRALEI DEFINITE
20 PRINT "X0=";
30 INPUT D
40 PRINT "X2P="
50 INPUT E
60 PRINT "P=";
70 INPUT P
80 B=(E−D)/2/F
90 A=0
100 X=D
110 GOSUB 500
120 A=Y+A
130 X=X+B
140 GOSUB 500
150 X=4*Y+A
160 X=X+B
170 GOSUB 500
180 A=Y+A
190 F=F−1
200 IF F=0 THEN 220
210 GO TO 40
220 C=A*B/3
230 PRINT "REZ=" ; C
240 STOP
500 Y=((X−2)*X−1)*X+2
510 RETURN
```

EDITURA TEHNICĂ

(continuare de la vol. 1)

• Un număr la fel de important de programe, din toate domeniile de aplicație, rezolvă probleme simple și complexe de matematică-fizică din industrie și învățămînt (aici este inclus și un îndrumar pentru predarea matematicii asistate de calculatorul personal); alte programe rezolvă jocuri, trasează grafice și figuri etc.

• Specialiștilor le atragem atenția asupra existenței comenzilor pentru toate tipurile de monitoare aMIC și – în acest context – asupra expunerii complete a microprocesorului Z80, cum și a uneia succinte a microprocesoarelor 8080.

• Sînt incluse programe de testare și indicații de depanare, ca și referiri concrete la perifericele utilizabile cu aMIC – display-uri (televizoare SPORT, de exemplu), casete ș.a.

• Editura Tehnică se mîndrește cu activitatea sa de pionierat în sprijinirea introducerii intensive a calculatoarelor personale românești și recomandă cu căldură consultarea acestei cărți din noul ciclu „Totul despre...", ca și a altor cărți de informatică, inclusiv a volumelor din seria continuă AMC (Automatică–Management–Calculatoare).

Vol. I și II Lei 42

